HORIZON: A Benchmark for in-the-wild User Behavior Modelling

Arnav Goel[1†] **Pranjal A. Chitale**[2] **Bhawna Paliwal**[3†]
Bishal Santra[2] **Amit Sharma**[2]
[1]Carnegie Mellon University [2]Microsoft Research India [3]University of California, Berkeley
Correspondence: amshar@microsoft.com
[†]**Work done while at Microsoft Research India.**

Abstract

User-interaction sequences in modern recommendation systems often showcase complex temporal patterns that pose fundamental challenges in time series modeling. However, existing user modeling approaches consider benchmarks mostly focused on short-term, single-domain next-item prediction, employing in-distribution evaluation practices that fail to assess true temporal / cross-user OOD generalization capabilities. Furthermore, most benchmarks use leave-one-out or ratio-based splits that risk temporal leakage and reward models which exploit short-range correlations rather than capturing evolving user preferences. We introduce HORIZON, a large-scale benchmark designed to establish robust evaluation practices for sequential recommendation and user behavior modeling. Built as a cross-domain reformulation of Amazon Reviews Dataset, it covers 54M users, 35M items, and 486M interactions, enabling both pre-training as well as rigorous out-of-distribution evaluations. HORIZON tests three core capabilities essential for real-world deployment: (i) long-term temporal generalization, (ii) cross-domain transfer, and (iii) unseen-user generalization for cold-start settings. Our results demonstrate that while traditional baselines (e.g., BERT4Rec) perform well in-domain, they significantly degrade under temporal and unseen-user scenarios, and even state-of-the-art LLMs struggle in this task. Our findings underscore the gap between current models and the complex temporal, cross-domain nature of real-world user behavior.

1 Introduction

Personalization has become a cornerstone of digital platforms ranging from e-commerce and streaming services to news feeds and smart-home systems. At the core of personalization lies the task of *user modeling* constructing representations of individual behavior based on previous interactions. This task shares fundamental characteristics with time series modeling: understanding temporal dependencies, handling long-range patterns, and generalizing across different contexts.

However, user modeling has historically been studied through single-domain benchmarks such as MovieLens [1] and Amazon Reviews [2], where the focus is on predicting the next item in a short session. While effective for early recommendation research, such setups fall far short of capturing the complexity of modern user behavior.

In practice, user histories are long, sparse, and multi-faceted: individuals interact with diverse content types (e.g., books, electronics, clothing) and display evolving interests that unfold over months or years. Benchmarks confined to a single domain and short horizons therefore encourage models to lean heavily on item–item similarities or short-range correlations, rather than uncovering deeper semantic patterns necessary for understanding long-term preferences across domains. As a result, they fail to test whether models *generalize* (1) temporally, (2) across domains, or (3) to new users.

NeurIPS 2025 Workshop on Recent Advances in Time Series Foundation Models (BERT^{2}S).

Recent progress in sequential recommendation has introduced increasingly powerful architectures in transformers [3, 4], pre-trained large language models (LLMs) [5, 6], and, more generally complex models capable of handling long histories. Yet, without sufficiently comprehensive benchmarks, the true capacity of these models to capture evolving, cross-domain behavior remains unclear. This gap mirrors the broader challenge faced across machine learning: determining whether large pre-trained models are genuinely learning transferable temporal representations, or simply overfitting to narrow supervised tasks.

In this work we present HORIZON, a large-scale benchmark explicitly designed to address these limitations. Constructed from a cross-domain reformulation Appendix A of Amazon Reviews, HORIZON comprises 54M users, 35M items, and nearly 500M timestamped interactions, enabling both large-scale pretraining and rigorous downstream evaluation. Unlike prior benchmarks, HORIZON introduces evaluation protocols reflecting real-world deployment settings: (i) **temporal generalization** across long horizons, (ii) **cross-domain transfer** between heterogeneous content types, and (iii) **unseen-user adaptation** under cold-start or out-of-distribution conditions.

By framing personalization as a temporally evolving, multi-domain sequence modeling task, HORIZON connects recommendation research to broader questions about generalization and transferability raised by foundation models. While our core focus is user modeling, the methodological overlap with time series learning is direct: understanding how models extend beyond local patterns to capture long-range dynamics. Our initial experiments reveal that even state-of-the-art sequential recommender models struggle under these new conditions, underscoring the need for development of superior training paradigms which improve generalization.

2 Proposed Evaluation Framework and Task formulations

2.1 Limitations of Traditional Evaluation

Most existing recommendation benchmarks rely on **in-distribution** evaluation, where training, validation, and test splits are sampled from the same user traces. The two standard protocols are: *Leave-One-Out*, which holds out the last interaction per user, and *Ratio-Based Splits*, which partition sequences by a fixed 8:1:1 ratio. Both approaches risk temporal leakage across splits and offer no mechanism to test generalization to new users or across longer horizons [7, 8]. As a result, models are rewarded for exploiting short-range correlations rather than capturing evolving preferences.

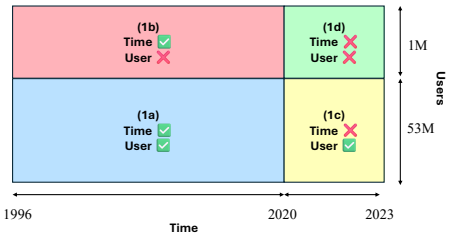

Figure 1: Proposed evaluation splits on the HORIZON benchmark for Task 1.

2.2 Proposed Evaluation Framework

To address temporal generalization, we introduce a **time-based cutoff protocol** that separates training and evaluation by a global timestamp τ, ensuring strict temporal fidelity. HORIZON's cross-domain construction from Amazon Reviews spans diverse product categories, inherently creates natural distribution shifts that test model transferability. Additionally, we hold out a subset of users exclusively for evaluation, enabling explicit measurement of out-of-distribution (OOD) generalization under cold-start conditions.

This yields four complementary evaluation scenarios (Fig. 1):

- **(1a) In-Domain, Aligned:** Leave-One-Out for training users before τ, reflecting short-horizon, in-distribution prediction.
- **(1c) In-Domain, Extrapolation:** Evaluation on all post-τ interactions of training users, probing long-range temporal generalization.
- **(1b) OOD-User, Aligned:** Leave-One-Out on held-out users before τ, testing adaptation to unseen user identities.
- **(1d) OOD-User, Extrapolation:** Predicting all post-τ interactions for unseen users, the most challenging setting combining user- and time-shift.

Table 1: In-Distribution - Temporally Aligned Evaluation (N=NDCG, M=MRR, R=Recall)

Baseline	N			M			R		
	10	50	100	10	50	100	10	50	100
CORE	8.5	8.7	8.7	7.25	7.30	7.30	12.1	13.0	13.4
SASRec	25.2	27.4	27.9	22.5	22.9	23.0	34.1	43.8	46.6
BERT4Rec	26.4	27.8	28.2	23.9	24.2	24.3	33.9	40.4	42.9
GRU4Rec	0.08	0.12	0.14	0.07	0.07	0.08	0.14	0.31	0.43

Table 2: OOD - Temporally Aligned Evaluation (N=NDCG, M=MRR, R=Recall)

Baseline	N			M			R		
	10	50	100	10	50	100	10	50	100
CORE	5.9	6.8	7.2	4.19	4.39	4.43	11.1	15.4	17.9
SASRec	17.8	19.2	19.6	15.2	15.5	15.5	26.2	32.2	34.6
BERT4Rec	11.8	14.4	15.2	9.96	10.50	10.58	17.8	29.5	34.7
GRU4Rec	0.01	0.01	0.02	0.004	0.004	0.005	0.01	0.03	0.08

Table 3: In-Distribution - Temporal Extrapolation Evaluation (N=NDCG, M=MRR, R=Recall)

Baseline	N			M			R		
	10	50	100	10	50	100	10	50	100
CORE	0.09	0.47	0.75	0.04	0.11	0.13	0.26	2.10	3.78
SASRec	2.9	3.6	3.9	1.88	2.03	2.05	6.2	9.4	11.0
BERT4Rec	1.1	3.2	4.0	0.56	0.99	1.10	2.8	12.8	17.8
GRU4Rec	0.01	0.01	0.02	0.004	0.005	0.01	0.01	0.03	0.08

Table 4: OOD - Temporal Extrapolation Evaluation (N=NDCG, M=MRR, R=Recall)

Baseline	N			M			R		
	10	50	100	10	50	100	10	50	100
CORE	0.10	0.53	0.82	0.04	0.12	0.15	0.32	2.33	4.13
SASRec	3.1	3.9	4.1	2.01	2.17	2.19	6.7	9.9	11.6
BERT4Rec	1.1	3.4	4.3	0.55	1.02	1.10	2.8	13.7	18.9
GRU4Rec	0.01	0.01	0.02	0.004	0.004	0.005	0.01	0.04	0.07

This four-way split disentangles temporal vs. user generalization, offering a more realistic testbed for sequential and foundation-style models.

2.3 Task Formulation

We propose the following task formulation to evaluate traditional and LLM based user modeling systems on HORIZON:

Task 1 — Sequential Next-Item Prediction. Traditional ID-based sequential recommendation using the four-way evaluation protocol above. This establishes baselines for temporal and cross-domain generalization using established architectures.

Task 2 — Generative Next-Item Prediction. Generative models like LLMs reformulate user histories into diverse search queries $Q = \{q_1, \ldots, q_{10}\}$ capturing multi-faceted user intent. Queries and catalog items are embedded into shared semantic space; an ANN index retrieves top-K candidates per query for final recommendation ranking. Figure 6 demonstrates the detailed pipeline used for the evaluation process.

Task 3 — Long-Horizon Behavior Modeling. User modeling requires capturing longer-term behavior patterns over extended time windows [9, 10]. We propose long-horizon modeling on the HORIZON benchmark, leveraging longer cross-domain user histories. Given user interaction history prior to a temporal cutoff τ., the generative model generates natural language descriptions of the next 10 likely engagement items, representing high-level future behavior summaries. Using the same retrieval pipeline (Figure 6), each description is embedded to retrieve matching catalog items.

3 Results & Discussion

3.1 Benchmarking traditional ID-based baselines

Tables 1 to 4 demonstrate the performance of traditional ID-based baselines across both In-Distribution as well as of Out-of-distribution settings across both temporal alignment as well as extrapolation setups.

Challenging Nature of the Task Unlike prior benchmarks on narrow domains (e.g., Beauty in [11]), HORIZON spans the full distribution of user activity across diverse product categories with 35M items. This multi-domain setting proves considerably more challenging: simple RNN-based models such as GRU4REC—which perform well in smaller setups [12]—struggle here, while attention-based models (BERT4REC, SASREC) prove more effective, underscoring the need for flexible context modeling in heterogeneous histories.

Table 5: Generative Next-Item Prediction

Model	Recall			Precision		
	@10	@50	@100	@10	@50	@100
LLAMA-3.1-8B	1.62	2.37	2.84	0.20	0.23	0.22
Qwen3-8B	2.06	2.95	3.50	0.25	0.28	0.28
Gemma2-9B	1.45	2.26	2.66	0.16	0.21	0.19

Table 6: Long-Horizon Behavior Modeling

Model	Recall			Precision		
	@10	@50	@100	@10	@50	@100
LLAMA-3.1-8B	1.26	6.52	13.25	0.51	0.52	0.53
Qwen3-8B	1.51	7.78	15.75	0.63	0.65	0.66
Gemma2-9B	0.98	5.07	10.39	0.42	0.43	0.44

Traditional Evaluation Overestimates Robustness Standard in-domain evaluation (Table 1) shows strong performance, creating false confidence in model capabilities. However, when tested on entirely unseen users (Table 2), performance drops sharply across all methods. The severity of performance drops is lower in case of attention-based models. This systematic overestimation of robustness validates HORIZON's OOD-based evaluation design and highlights critical gaps in current benchmarking practices.

Temporal Drift Exposes Fundamental Model Limitations Tables 3 and 4) causes catastrophic performance degradation across all methods. Critically, models generalize better to *new users within the same timeframe* than to *the same users across distant horizons*. This reveals heavy reliance on ID-based representations that fail when new items emerge without semantic grounding — a fundamental limitation for production systems facing evolving catalogs as time progresses.

3.2 Benchmarking Query Reformulation-based Generative Recommendation

Table 5 reports results for three prominent LLMs—LLAMA-3.1-8B, Qwen3-8B, and Gemma2-9B—on reformulating user histories into queries for item retrieval. Overall, performance is modest: Recall and Precision improve with larger candidate sets ($10 \rightarrow 100$), suggesting LLMs capture fragments of user intent but struggle with consistent accuracy. Qwen3-8B outperforms the others across metrics. To assess semantic quality, we measured similarity between reformulated queries and ground-truth items using BLAIR embeddings. Average cosine scores (0.71–0.73) indicate that queries are reasonably related but not sharply aligned, leaving room for more targeted reformulation. We conducted fine-tuning experiments using parameter-efficient (LoRA) and full fine-tuning approaches with LLaMA-3.1-8B and Qwen3-8B models, demonstrating similar trends presented in Appendix F.

3.3 Benchmarking Long-Horizon Generative Recommendation

Table 6 summarizes LLM results on predicting user interests beyond immediate interactions. Here, Recall@K improves with larger k, showing that models capture some relevant signals across extended horizons, but Precision remains low, reflecting a high rate of irrelevant predictions. As in query reformulation, Qwen3-8B consistently leads. Importantly, long-horizon tasks benefit from multiple valid future targets (unlike Task 2), which partly inflates Recall. Prior work [10, 9] has also relaxed strict temporal ordering, complicating direct comparisons. Thus, while results suggest some preference evolution modeling, current approaches to long horizon modeling remain limited in precision and robustness across tasks.

4 Conclusion

We identified a critical gap between real-world deployment requirements and existing time series user modeling benchmarks, which fail to test temporal generalization, cross-domain transfer, and cold-start adaptation. To address this, we presented HORIZON, a novel benchmark with five evaluation setups that systematically test models' generalization capabilities across out-of-distribution users, temporal settings, and long-horizon scenarios. Our extensive experiments across multiple models revealed significant performance discrepancies in different generalization scenarios that traditional benchmarks fail to capture. These findings strongly suggest that future research on sequential recommendation should prioritize measuring temporal complex behavior modeling capabilities rather than focusing exclusively on next-item prediction performance, and explicitly account for the generalization discrepancies we observed in unseen users and temporal extrapolation scenarios. Thus, the HORIZON benchmark aims to drive progress toward more robust user behavior modeling techniques enabling more personalized/engaging user experiences in modern web platforms.

References

[1] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

[2] Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 188–197, 2019.

[3] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 1441–1450, New York, NY, USA, 2019. Association for Computing Machinery.

[4] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 197–206, 2018.

[5] Zhenrui Yue, Sara Rabhi, Gabriel de Souza Pereira Moreira, Dong Wang, and Even Oldridge. Llamarec: Two-stage recommendation using large language models for ranking. *arXiv preprint arXiv:2311.02089*, 2023.

[6] Yuling Wang, Changxin Tian, Binbin Hu, Yanhua Yu, Ziqi Liu, Zhiqiang Zhang, Jun Zhou, Liang Pang, and Xiao Wang. Can small language models be good reasoners for sequential recommendation? In *Proceedings of the ACM Web Conference 2024*, pages 3876–3887, 2024.

[7] Duncan McElfresh, Sujay Khandagale, Jonathan Valverde, John Dickerson, and Colin White. On the generalizability and predictability of recommender systems. *Advances in Neural Information Processing Systems*, 35:4416–4432, 2022.

[8] Yitong Ji, Aixin Sun, Jie Zhang, and Chenliang Li. A critical study on data leakage in recommender system offline evaluation. *ACM Transactions on Information Systems*, 41(3):1–27, 2023.

[9] Zhihan Zhou, Qixiang Fang, Leonardo Neves, Francesco Barbieri, Yozen Liu, Han Liu, Maarten W Bos, and Ron Dotsch. Use: Dynamic user modeling with stateful sequence models. *arXiv preprint arXiv:2403.13344*, 2024.

[10] Nikil Pancha, Andrew Zhai, Jure Leskovec, and Charles Rosenberg. Pinnerformer: Sequence modeling for user representation at pinterest. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 3702–3712, New York, NY, USA, 2022. Association for Computing Machinery.

[11] Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. Bridging language and items for retrieval and recommendation. *arXiv preprint arXiv:2403.03952*, 2024.

[12] Filippo Betello, Antonio Purificato, Federico Siciliano, Giovanni Trappolini, Andrea Bacciu, Nicola Tonellotto, and Fabrizio Silvestri. A reproducible analysis of sequential recommender systems. *IEEE Access*, 13:5762–5772, 2025.

[13] Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, et al. Mind: A large-scale dataset for news recommendation. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 3597–3606, 2020.

[14] Wei Jin, Haitao Mao, Zheng Li, Haoming Jiang, Chen Luo, Hongzhi Wen, Haoyu Han, Hanqing Lu, Zhengyang Wang, Ruirui Li, et al. Amazon-m2: A multilingual multi-locale shopping session dataset for recommendation and text generation. *Advances in Neural Information Processing Systems*, 36:8006–8026, 2023.

[15] Wei Jin, Haitao Mao, Zheng Li, Haoming Jiang, Chen Luo, Hongzhi Wen, Haoyu Han, Hanqing Lu, Zhengyang Wang, Ruirui Li, Zhen Li, Monica Cheng, Rahul Goutam, Haiyang Zhang, Karthik Subbian, Suhang Wang, Yizhou Sun, Jiliang Tang, Bing Yin, and Xianfeng Tang. Amazon-m2: a multilingual multi-locale shopping session dataset for recommendation and text generation. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

[16] Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, and Ming Zhou. MIND: A large-scale dataset for news recommendation. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3597–3606, Online, July 2020. Association for Computational Linguistics.

[17] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Yujie Lu, Hui Wang, Changxin Tian, Yingqian Min, Zhichao Feng, Xinyan Fan, Xu Chen, Pengfei Wang, Wendi Ji, Yaliang Li, Xiaoling Wang, and Ji-Rong Wen. Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms. In *CIKM*, pages 4653–4664. ACM, 2021.

[18] Wayne Xin Zhao, Yupeng Hou, Xingyu Pan, Chen Yang, Zeyu Zhang, Zihan Lin, Jingsen Zhang, Shuqing Bian, Jiakai Tang, Wenqi Sun, Yushuo Chen, Lanling Xu, Gaowei Zhang, Zhen Tian, Changxin Tian, Shanlei Mu, Xinyan Fan, Xu Chen, and Ji-Rong Wen. Recbole 2.0: Towards a more up-to-date recommendation library. *arXiv preprint arXiv:2206.07351*, 2022.

[19] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks, 2016.

[20] Yupeng Hou, Binbin Hu, Zhiqiang Zhang, and Wayne Xin Zhao. Core: Simple and effective session-based recommendation within consistent representation space. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 1796–1801, New York, NY, USA, 2022. Association for Computing Machinery.

[21] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[22] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025.

[23] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozińska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucińska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju yeong Ji, Kareem

Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Cogan, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. Gemma 2: Improving open language models at a practical size, 2024.

[24] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library. *arXiv preprint arXiv:2401.08281*, 2024.

Attribute	PF	Amz-M2	MIND	Amz-Reviews	HORIZON
No. of users	N/A	N/A	1M	54.51M	**54.51M**
Avg User History Length	N/A	4.2	N/A	3.86	**9.07**
No. of items	N/A	1.42M	0.16M	34.52M	**34.52M**
No. of interactions	N/A	16.78M	24.15M	485.89M	**485.89M**
Cross-domain	✓	✓	×	×	✓
Diversity	✓	×	×	×	✓
Interaction Timestamps	–	×	×	✓	✓
Open-Source	×	✓	✓	✓	✓

Table 7: Comparison of existing Sequential Recommendation Benchmarks with HORIZON. (PF refers to PinnerFormer, Amz-M2 refers to Amazon M2, Amz-Reviews is the Amazon Reviews dataset.

A Benchmark and Benchmark Stats/Comparison with Existing Benchmarks

Benchmark Description: User modeling and sequential recommendation aim to predict a user's future interactions based on their past behavior. Formally, for a user u, we observe a sequence of interactions over time $\mathcal{H}_u = [i_1, i_2, \ldots, i_t]$, where i_t denotes the item interacted with at time t. The objective is to estimate the likelihood of the next interaction i_{t+1} or future next interactions over some time period T $i.e.$ $\hat{i}_{t+1,..,T} = (i_{t+1}, i_{t+2}, ...i_T)$, given the user's historical context:

$$\hat{i}_{t+1} = \arg\max_{i \in \mathcal{I}} \Pr(i \mid \mathcal{H}_u),$$

where $\mathcal{I}$ denotes the candidate item set. This formulation underpins several established benchmarks such as MIND, M2, and Amazon Reviews [13, 14, 11]. As noted in Section 2, the Amazon Reviews dataset has become a widely used resource for training and evaluating sequential recommenders. However segregates user interactions by product categories, making it domain-specific and thus limiting its ability to capture holistic user preferences. In the real world, users engage with a variety of domains, and isolating interactions to a single domain introduces artificial boundaries, resulting in incomplete modeling of cross-domain behaviors and potentially spurious patterns causing incorrect user modeling.

To address this limitation, we introduce HORIZON, a large-scale benchmark designed to support cross-domain user modeling and sequential recommendation. HORIZON is constructed by refactoring and consolidating the Amazon Reviews 2023 dataset [11], merging interactions across all available categories to create unified, realistic user histories. The resulting benchmark comprises of 53.5 million users and 34.5 million unique items, enabling rigorous evaluation of models under settings that better reflect real-world recommendation scenarios.

A.1 Comparison with Existing Benchmarks

Table 7 provides a comparative analysis of our dataset against existing sequential recommendation benchmarks. While proprietary datasets like PinnerFormer [10] offer scale and diversity, they remain inaccessible to the broader research community. Public datasets such as Amazon-M2 [15] provide cross-domain capabilities but lack temporal depth due to being being restricted to session-based interactions rather than long-term user modeling. The MIND dataset [16], despite its million-user scale, covers only two weeks of user history, severely limiting its utility for long-horizon recommendation research. Similarly, the Amazon Reviews dataset [2, 11] provides timestamps but artificially segments interactions into isolated domains. In contrast, HORIZON uniquely combines cross-domain coverage, interaction diversity, and comprehensive temporal information, enabling more realistic evaluation of sequential recommendation systems across extended time horizons.

B HORIZON Statistics and Plots

The HORIZON benchmark is curated by reformulating the widely-used Amazon Reviews 2023 dataset [11], merging all 33 categories into unified user histories to enable robust long-term, cross-domain user modeling. This section provides an in-depth statistical analysis of the dataset through visualizations and derived insights.

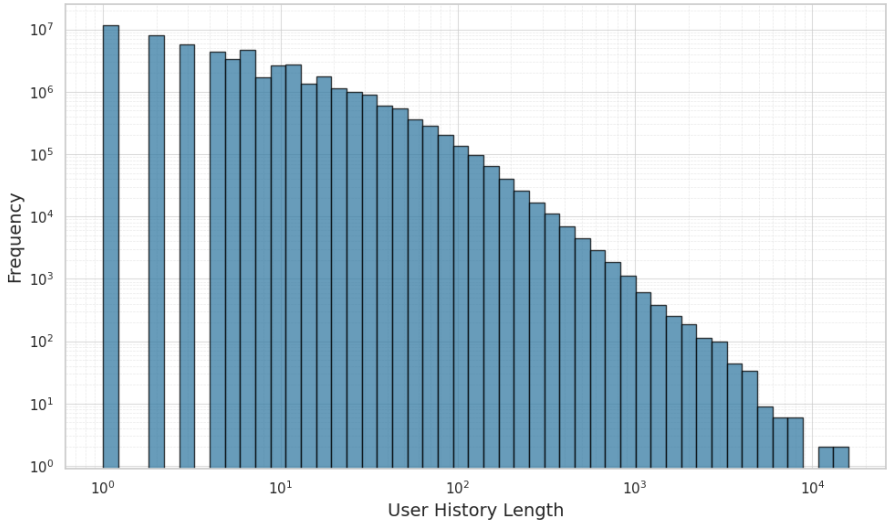

Figure 2: Histogram Depicting the Frequency Distribution of User History Lengths in HORIZON. The presence of ultra-long user histories highlights the need for architectures capable of modeling long-range sequential dependencies.

Scale and Diversity: The benchmark comprises approximately **53.5M users and 34.5M unique items**, amounting to nearly 486M interaction records. This scale is significantly larger than prior public benchmarks and captures highly diverse behavioral patterns. With the unified formulation, user histories naturally span multiple product categories—introducing heterogeneous context that is both semantically diverse and temporally rich. This setting reflects real-world personalization challenges more faithfully than isolated category-based modeling.

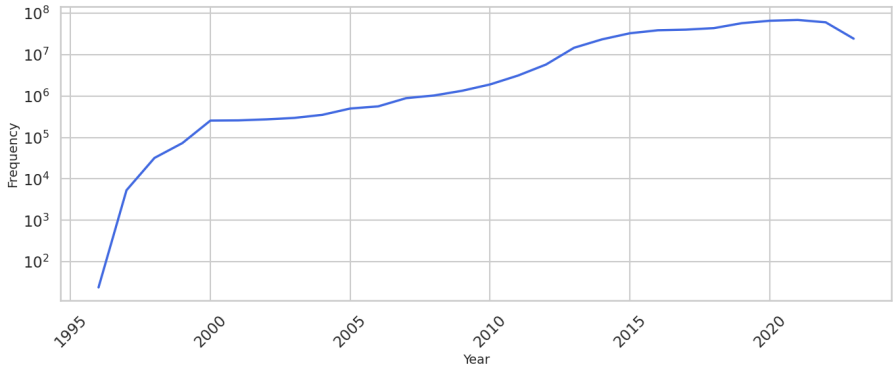

Figure 3: Line Plot Depicting the Temporal Distribution of User Histories in HORIZON. The balanced volume before and after 2020 makes it suitable for temporal extrapolation tasks.

User History Lengths: Figure 2 illustrates a long-tailed distribution of user history lengths in HORIZON. While a large portion of users exhibit short interaction sequences, there exists a substantial number with extremely long histories—extending beyond 1,000 timestamps for tens of thousands of users. This highlights the need for models capable of handling long-range dependencies and memory-efficient representations. Traditional sequence models struggle in this regime due to vanishing gradients and computational bottlenecks, motivating the exploration of transformer-based or memory-augmented architectures for this benchmark.

Temporal Structure and Generalization: The temporal distribution of interactions (Figure 3) reveals a sharp rise in user activity post-2010, peaking around 2020. Crucially, nearly half the

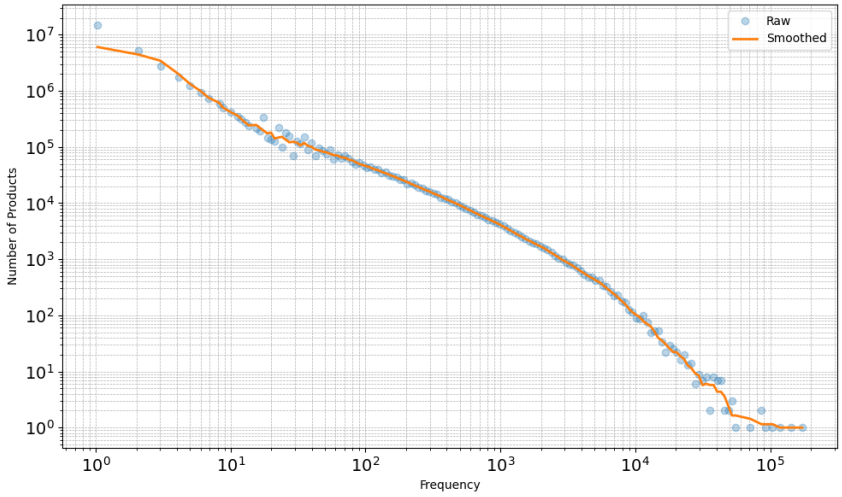

Figure 4: Frequency Distribution of Products in the HORIZON Benchmark. The power-law structure reflects extreme item sparsity, with most items having very few interactions.

interactions occur after the 2020 temporal cut-off used in our evaluation framework. Specifically, the average number of timestamps before 2020 is 4.99, while it remains comparable after 2020 at 4.09. This temporal balance ensures that both training and test splits are adequately rich, setting up a robust testbed for extrapolative evaluation and temporal generalization. As models are evaluated on unseen user interactions post-2020, they are challenged to infer future behavior patterns from past, potentially outdated, preferences—mirroring real-world drift in user intent.

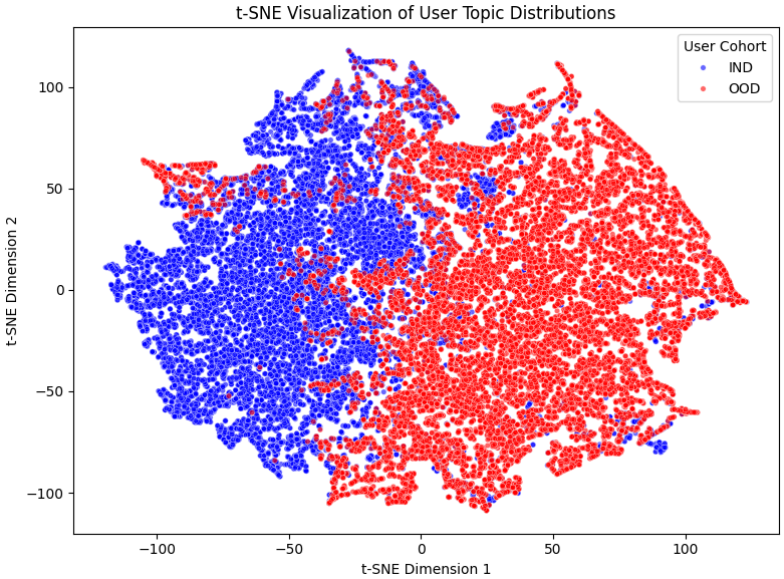

Figure 5: t-SNE depicting the distinct user topic distributions in the in-distribution and OOD users.

Product Distribution: Figure 4 plots the frequency distribution of product IDs, which exhibits a pronounced long-tail trend. A small fraction of items dominate interactions, while the majority are sparsely interacted with. This reflects typical e-commerce dynamics but poses unique challenges for recommender systems: most prior models are biased toward frequent items. The high item cardinality (34M) and sparse tail necessitate models that generalize well to rarely seen or previously unseen

products. Incorporating textual features or content-based augmentations could be beneficial in this context, especially under cold-start settings.

Benchmark Design Implications: The three key observations from these plots underscore the difficulty of the HORIZON benchmark:

1. **Long Histories:** Users with thousands of interaction points require models that capture dependencies over extended horizons and adapt across evolving interests.
2. **Temporal Drift:** A significant portion of test data lies beyond the training horizon (post-2020), enforcing extrapolation beyond the training distribution and testing robustness to temporal shifts.
3. **Item Sparsity:** The skewed product frequency implies that many test items are low-frequency or unseen, further intensifying the generalization challenge.

Taken together, HORIZON enables a comprehensive stress test of user behavior models across multiple axes—scale, history length, temporal generalization, and sparsity. Its unified multi-category formulation fosters the development of general-purpose, temporally robust, and cross-domain recommendation architectures.

C Task 1 Splits and Out-of-Distribution Analysis

In our proposed Task 1 setup, the user population is explicitly partitioned into two cohorts to rigorously test generalization: *in-distribution (IND)* users observed during training, and *out-of-distribution (OOD)* users who are entirely held out. The fixed temporal cutoff at $\tau = 2020$ allows us to decouple user generalization from temporal extrapolation. Below, we elaborate on the statistical and structural distinctions between these cohorts, which underline the difficulty of the proposed evaluation.

Temporal Shift and Behavioral Drift: As visualized in Figure 3, a significant volume of user activity in the dataset occurs post-2020. By construction, OOD users are sampled from this post-2020 pool, whereas IND users have interactions both before and after the temporal boundary. This creates a natural distributional shift: the OOD cohort is inherently more recent and behaviorally different, reflecting newer products, evolving user preferences, and potentially different session structures. Hence, even under temporally aligned evaluation (Subtask 1c), the OOD test set exhibits non-trivial variance from the training distribution.

Semantic Divergence via Topic Modeling. To investigate the semantic distinctiveness between in-distribution (IND) and out-of-distribution (OOD) user groups, we apply Latent Dirichlet Allocation (LDA) to model topics from user review histories, treating each user as a document composed of concatenated item descriptions and metadata. The resulting topic distributions uncover meaningful divergence in user interests. Both groups engage with broad product themes (e.g., books, electronics, fashion), yet OOD users demonstrate stronger focus on niche and emergent categories. For example, OOD-specific topics include terms like *"telescope," "kite," "bjj," "freemason,"* and musical instruments such as *"guitar," "ukulele," "pedal"*, suggesting a shift toward specialized or subcultural interests. In contrast, IND topics reflect more mainstream and diversified engagement, including wellness supplements (e.g., *"nootropic," "creatine," "arginine"*) and general home goods. To quantify these patterns, we compute entropy and dominance over user topic distributions. OOD users show significantly lower entropy (mean = 1.18 vs. 1.28) and higher topic dominance (mean = 0.51 vs. 0.48), indicating more focused topical preferences. A t-SNE projection of user topic vectors reveals clear separation between IND and OOD clusters. Additionally, the average KL divergence from IND to OOD topic distributions exceeds 0.8, reinforcing the semantic shift. These findings suggest that OOD generalization reflects not just temporal drift but substantive thematic changes in user behavior and product engagement.

D Experimental Setup

Task 1 Setup: We adopt a temporal cut-off of $\tau = 2020$ to define the training window. From the full dataset of $\sim$54M users, we randomly sample 1M users which atleast have post-τ interactions as our out-of-distribution (OOD) user set, and treat the remaining 53M as the in-distribution (IND)

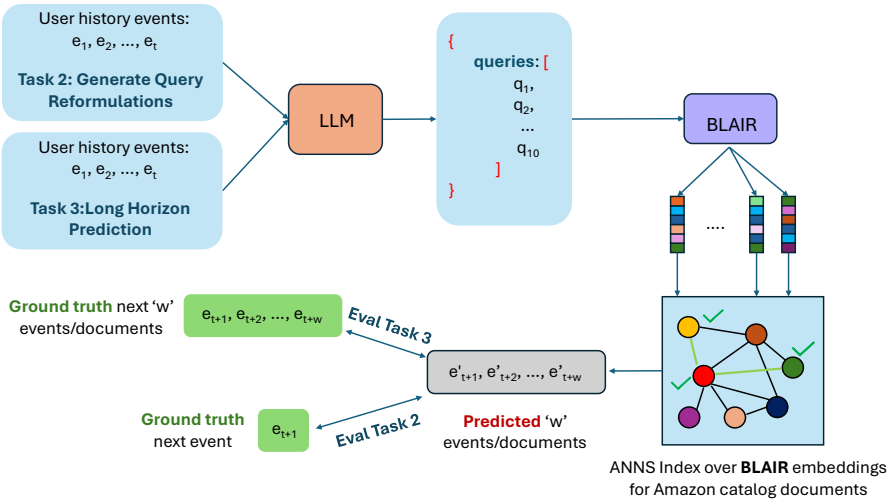

Figure 6: Pipeline Detailing the LLM Generation, Retrieval and Evaluation Process Proposed for Tasks 2 and 3.

pool. From this IND pool, 1M users are sampled to construct the test set for sub-task (1c). Due to computational constraints, we train all models on a 100K user subset of the IND set, and evaluate on 25K users each for sub-task (1d) (IND extrapolation) and sub-task (1c) (OOD prediction). For all baselines, we use the RECBOLE framework [17, 18], which offers standardized implementations and reproducible pipelines for recommendation models. The following popular item-ID-based baselines are included:

Models and Setup: GRU4REC [19] employs a recurrent architecture with gated recurrent units to capture sequential dependencies in user histories. SASREC [4] adopts a transformer-based architecture with self-attention mechanisms to model user behavior sequences. BERT4REC [3] utilizes a bidirectional transformer encoder trained with a Cloze-style objective to leverage full-sequence context. CORE [20] formulates session representations as weighted linear combinations of item embeddings, aligning both session and item vectors in a shared latent space.

Evaluation Metrics: While these methods are typically evaluated using either ratio-based or leave-one-out strategies, we retrain and evaluate them under the temporally grounded evaluation protocol described in Figure 1. All models are trained with standardized hyperparameters and evaluated on our four evaluation settings using **MRR**, **Recall@K**, and **NDCG@K** for $K = \{10, 50, 100\}$.

Task 2 and 3 Setup:

For Tasks 2 and 3, we use the held-out out-of-distribution (OOD) test set comprising 1M users as our evaluation benchmark. We primarily focus on evaluating the zero-shot capabilities of LLMs for modeling user behavior, as effective training paradigms for LLMs in recommendation settings remain an open research problem and present unique challenges in our context given the extremely long-tailed item distribution. Nevertheless, we include standard fine-tuning baselines (PEFT and full fine-tuning) for completeness.

Models and Setup: We evaluate three recent and publicly available language models up to 9B parameter scale: LLAMA-3.1-8B [21], QWEN3-8B [22], and GEMMA2-9B [23]. All models are queried in a zero-shot manner using a standardized prompt for each task.

Retrieval Pipeline: For encoding the items and queries, we the use the pre-trained BLAIR item encoder [11] as it is pre-trained on the Amazon-Reviews items and the FAISS library [24] for creating the ANN-based vector databases to perform retrieval. An approximate nearest neighbor (ANN) index is constructed over catalog item embeddings $\{\mathbf{i}_j\}$, and top-K candidates are retrieved for each query embedding $\mathbf{q}_k$. These are merged to form a final set of K recommendations $\hat{I}$.

Evaluation Metrics: As we do not perform ranking across queries, we compute standard retrieval metrics i.e. RECALL@K and PRECISION@K for $K = 10, 50, 100$ to assess the effectiveness of the generated outputs in retrieving relevant items.

E Hyperparameters and Implementation Details

E.1 RecBole Experiments - Task 1

All models in Task 1 were trained using the RecBole framework [17, 18] with a consistent configuration to ensure a fair comparison. The common training hyperparameters were selected based on prior literature and empirical tuning on a held-out validation set. These include a small learning rate of 2×10^{-5} to stabilize optimization over long sequences, a maximum of 10 epochs for training, and early stopping with a patience of 10 epochs to prevent overfitting. To ensure reproducibility across all experimental runs, we fixed the random seed to 2025.

Training Hyperparameters. All models were trained with the following consistent configuration

- **Learning rate**: 2×10^{-5}
- **Maximum epochs**: 10
- **Early stopping patience**: 10
- **Random seed**: 2025
- **Maximum sequence length**: 100
- **Validation metric**: MRR@10
- **Evaluation cutoffs**: $k \in \{10, 20, 50, 100\}$
- **Test negative samples**: 100

To support uniform evaluation across models, we truncated all user interaction sequences to a maximum of 100 items and used mean reciprocal rank at cutoff 10 (MRR@10) as the primary validation metric. During testing, we sampled 100 negative items for each user-item query to simulate realistic top-k recommendation settings and report metrics at various cutoffs (k).

Table 8: Model-specific hyperparameter configurations

Parameter	BERT4Rec	GRU4Rec	SASRec	CORE
Hidden/Embedding size	256	256	256	256
Number of layers	3	3	3	3
Attention heads	4	-	4	4
Dropout probability	0.15	0.15	0.15	0.15
Batch size	8192	8192	4096	4096
Loss function	BPR	BPR	CE	CE
Mask ratio	0.2	-	-	-

Model-Specific Hyperparameters Each model was configured using a 256-dimensional embedding and three layers to capture higher-order dependencies. Attention-based models (BERT4Rec, SASRec, and CORE) used 4 attention heads to balance modeling capacity and memory cost. A dropout rate of 0.15 was applied to all models for regularization. Batch sizes were tuned based on GPU memory availability and empirical training stability: 8192 for BERT4Rec and GRU4Rec, and 4096 for SASRec and CORE due to their higher per-batch memory footprint. These are further detailed in Table 8.

Architecture Details: Given below are the architectural details about the RecBole baselines which we have employed in our study on the `HORIZON` benchmark:

- **BERT4Rec**: It leverages bidirectional Transformers to model sequence-wide context and predicts masked items using a masked language modeling (MLM) objective, with a mask ratio set to 0.2.

- **GRU4Rec**: GRU4Rec uses gated recurrent units (GRUs) to model sequential dependencies.
- **SASRec**: SASRec is built on unidirectional self-attention layers, enabling it to capture short- and long-term dependencies without recurrence.
- **CORE**: CORE integrates self-attention with collaborative filtering signals, enhancing personalization through a hybrid architecture

Loss Function Configuration. Given below are the possible loss function configurations available in RecBole for training sequential recommendation models:

- **BPR models** (BERT4Rec, GRU4Rec, SASRec): Bayesian Personalized Ranking with negative sampling during training
- **CE models** (CORE): Cross-entropy loss without negative sampling during training

Models trained with BPR loss (BERT4Rec, GRU4Rec, SASRec) rely on dynamic negative sampling and optimize the ranking of positive over negative interactions. In contrast, CORE optimizes a classification objective using cross-entropy loss computed over the full softmax distribution.

Execution Details. All experiments were conducted using a high-performance compute cluster equipped with 4 NVIDIA A100 GPUs (80GB VRAM each). We employed PyTorch's automatic mixed precision (AMP) to accelerate training and reduce memory usage. Training time per epoch varied with architectural complexity: GRU4Rec, being lightweight, completed one epoch in approximately 0.75 hours, while BERT4Rec, with its attention-heavy encoder and MLM objective, required around 1.25 hours per epoch. Multi-GPU training was implemented using the NCCL backend for synchronized distributed training. All hyperparameters and implementation choices were fixed across all splits to ensure experimental consistency and comparability.

E.2 Task 2 and 3 Experiments

Table 9: Hyperparameters used for different models.

Hyperparameter	LLAMA-3.1-8B	QWEN3-8B	GEMMA2-9B
Batch Size	512	512	256
Temperature	0.7	0.7	0.7
Top-P	0.95	0.8	0.8
Top-K	-1	20	-1
Max-Tokens (Task 2)	220	220	220
Max-Tokens (Task 3)	350	350	350

LLM Inference Setup. We adopt a consistent inference pipeline for both Task 2 (LLM-based Next Product Recommendation via Query Reformulation) and Task 3 (LLM-based Long-Horizon User Modeling), as described in Section 5 and illustrated in Figure 2. All models are prompted in a zero-shot setting, without any fine-tuning or retrieval augmentation, to evaluate their general-purpose reasoning capabilities over long user histories.

We utilize three state-of-the-art, instruction-tuned open-source LLMs: LLAMA-3.1-8B [21], QWEN3-8B [22], and GEMMA2-9B [23]. These models were selected for their strong instruction-following capabilities and competitive performance on public benchmarks.

Table 9 summarizes the decoding hyperparameters used. The temperature was fixed at 0.7 across all models to balance determinism and diversity in outputs. We set Top-P and Top-K sampling parameters based on model-specific best practices to control generation randomness. The maximum token limits were adjusted per task: 220 tokens for Task 2 (shorter search queries), and 350 tokens for Task 3 (longer next-item descriptions). Batch sizes were selected based on each model's memory footprint and throughput on A100 GPUs, with the larger GEMMA2-9B model using a smaller batch size.

Execution Details. All inference was run using the vLLM engine on a compute cluster with 4× NVIDIA A100 40GB GPUs. The full test set consists of 1 million users, with each user processed independently in batched decoding mode. End-to-end inference across all models required approximately 5 days due to the volume of input prompts and the autoregressive nature of generation.

To support reproducibility and accessibility, we will release all evaluation code, prompt templates, and precomputed predictions on smaller held-out test splits post-acceptance. These subsets will enable low-resource experimentation on the same evaluation protocol without requiring access to large-scale GPU compute.

Generating Query and Item Embeddings using BLAIR. To encode the item catalog and predicted queries, we leverage the BLAIR item encoder [11], a RoBERTa-based model pretrained on Amazon review titles. We use the `hyp1231/blair-roberta-base` checkpoint via the HuggingFace Transformers library [1], and tokenize each product title with a maximum sequence length of 512 tokens. Embeddings are obtained by extracting the `[CLS]` token representation from the final hidden layer, followed by ℓ_2 normalization to facilitate cosine similarity-based retrieval. To scale embedding computation across a large number of titles, we utilize the `Accelerate` library with mixed-precision inference (fp16) and distributed processing across multiple GPUs, achieving efficient batch-wise encoding with a batch size of 4096. We shard the workload across processes and later merge the outputs to form a single embedding matrix for the catalog and prediction sets.

Retrieval and Indexing using FAISS. For approximate nearest neighbor (ANN) search, we employ the FAISS library [24], which implements the Hierarchical Navigable Small World (HNSW) graph-based indexing algorithm. We build a HNSW index on the catalog embeddings using cosine similarity as the distance metric. The key hyperparameters used during index construction include: `M=64`, which controls the number of bi-directional links created for each new node and influences index accuracy and memory usage; and `efConstruction=256`, which sets the dynamic list size for the graph during construction and affects indexing time and final recall quality. At query time, we use `efSearch=256` to control the breadth of the search and balance between latency and retrieval performance. These values were selected based on a grid search over the validation set to optimize top-k recall, where $k = 10$, while ensuring sub-millisecond retrieval latency per query on a modern GPU setup.

This setup enables scalable, low-latency nearest neighbor search over millions of product titles, while maintaining semantic alignment between predicted queries and candidate items.

F LLM-Finetuning baselines

Table 10: Comparison of Fine-tuned LLMs for Next-Item Prediction

Setting	Recall@K (%)			Precision@K (%)		
	FFT (LLaMA3)	LoRA (LLaMA3)	LoRA (Qwen3)	FFT (LLaMA3)	LoRA (LLaMA3)	LoRA (Qwen3)
In-Domain Temporal Extrapolation (Task 1c)						
K=10	1.45	1.65	1.38	0.98	1.29	0.90
K=50	1.67	1.82	1.60	0.97	1.28	0.90
K=100	2.02	2.09	1.93	0.97	1.28	0.89
Out-of-Domain Temporal Extrapolation (Task 1d)						
K=10	1.24	0.71	1.18	0.82	0.42	0.77
K=50	1.41	0.84	1.37	0.81	0.42	0.77
K=100	1.71	1.07	1.67	0.80	0.42	0.76

We observe that fine-tuned models (LLaMA-3.1-8B with both FFT and LoRA, and Qwen3-8B with LoRA), which generate only the next single item per user, achieve comparable performance to our zero-shot retrieval baseline setup described in Table 5 that generates 10 queries. The zero-shot approach is thus both simpler in execution and more scalable, especially as item catalogs grow.

Our findings highlight a key insight: standard instruction-tuning methods do not effectively exploit LLM capabilities in this long-tailed recommendation context. Unlike discriminative models that benefit from contrastive supervision and negative sampling (Task 1 results), LLM instruction-tuning tasks lack such structure. Future work should focus on novel training paradigms, such as contrastive

[1] `https://huggingface.co/hyp1231/blair-roberta-base`

losses with vocabularies explicitly aligned to item identifiers, which can help better exploit the representational and semantic power of LLMs in recommendation tasks.

G Prompts

G.1 Task 2: Query-Based Next-Item Recommendation.

Task 2 evaluates an LLM's ability to generate personalized search queries from a user's Amazon product history. The prompt asks the model to produce 10 queries—ranging from directly relevant to tangential and intentionally unrelated—balancing relevance with serendipity. These queries act as soft proxies for next-item prediction, revealing how well the model generalizes user intent. The setup is zero-shot, requiring the model to function as a semantic encoder-decoder without fine-tuning or examples.

```
PROMPT FOR TASK 2 - LLM-Based Next Item Recommendation:

You are an expert at turning a user's Amazon product history into personalized search queries.

History: I1 <SEP> I2 <SEP> ..... <In>
This was the users Amazon product history.

Your task is to generate a set of 10 personalized search queries that reflect the user's interests and
      preferences.
Try to balance diversity and serendipity with relevancy to the user history. These queries will be
     used to recommend the next product to the user.

Out of these 10 queries:
4 queries should be directly related to the user's history;
3 queries should be tangentially related;
3 queries should be completely unrelated but interesting.

Process:
1. Think of a guideline explaining what intents or aspects you observed in the user history which
     helped you formulate these queries. You don't need to specify which is which.
2. Then, generate exactly 10 search queries balancing core interests with a bit of serendipity.

## Output Format
Provide the response only as a JSON object with one field: (do not generate anything else)

{
  "queries": [
    "query1",
    "query2",
    "...",
    "query10"
  ]
}
```

G.2 Task 3

Following is the prompt for Task 3: Long-Horizon User Modeling using Large Language Models (LLMs). This task is designed to evaluate a model's ability to understand and extrapolate from a user's product history over time. The prompt guides the LLM to generate forward-looking, autoregressive item descriptions based on prior purchases, simulating realistic recommendation scenarios. Specifically, it instructs the model to infer underlying user preferences and behavioral patterns, and to generate coherent, temporally ordered predictions that balance relevance and serendipity. The prompt is framed in a zero-shot setting, encouraging the LLM to reason sequentially without access to explicit training examples.

```
PROMPT for Task 3 - LLM-Based Long-Horizon User Modeling:

You are an expert at predicting the next products a user may want based on their Amazon product
     history.

History: I1 <SEP> I2 <SEP> ..... <In>
This was the user's Amazon product history with exact product titles (NOT descriptions).

Your task is to generate descriptions for the next 10 items the user is most likely to be interested
     in. Provide concise, onesentence descriptions that capture the essence of each potential item.
     These will guide recommendation generation.

Try to model the sequences in the user history and provide a mix of relevant and serendipitous items
     trying to capture the user's interests, intents and changes in behavior. Use the first item
     description to guide your next timestep's item description generation in an autoregressive
     manner.

Process:
1. Think of a guideline explaining the patterns or preferences you observed in the user history that
     informed your item descriptions.
2. Provide exactly 10 next-item descriptions balancing relevance and serendipity generated one after
     the other in temporal order.

## Output Format
Provide the response only as a JSON object with one field: (do not generate anything else)

{
  "item_descriptions_timewise": [
    "item_description_time_step1",
    "item_description_time_step2",
    "...",
    "item_description_time_step10"
  ]
}
```