SHORT-PAPER

Legommenders: A Comprehensive Content-Based Recommendation Library with LLM Support

QIJIONG LIU, The Hong Kong Polytechnic University, Hong Kong, Hong Kong, Hong Kong

LU FAN, The Hong Kong Polytechnic University, Hong Kong, Hong Kong, Hong Kong

XIAOMING WU, The Hong Kong Polytechnic University, Hong Kong, Hong Kong, Hong Kong

Published: 08 May 2025

WWW '25: The ACM Web Conference 2025
April 28 - May 2, 2025
Sydney NSW, Australia

Conference Sponsors:
SIGWEB

https://doi.org/10.1145/3701716.3715305
ISBN: 9798400713316

LEGOMMENDERS: A Comprehensive Content-Based Recommendation Library with LLM Support

Qijiong Liu
The Hong Kong Polytechnic University
Hong Kong SAR
liu@qijiong.work

Lu Fan
The Hong Kong Polytechnic University
Hong Kong SAR
cslfan@comp.polyu.edu.hk

Xiao-Ming Wu*
The Hong Kong Polytechnic University
Hong Kong SAR
xiao-ming.wu@polyu.edu.hk

Abstract

We present LEGOMMENDERS[1], a unique library designed for content-based recommendation that enables the joint training of content encoders alongside behavior and interaction modules, thereby facilitating the seamless integration of content understanding directly into the recommendation pipeline. LEGOMMENDERS allows researchers to effortlessly create and analyze over 1,000 distinct models across 15 diverse datasets. Further, it supports the incorporation of contemporary large language models, both as feature encoder and data generator, offering a robust platform for developing state-of-the-art recommendation models and enabling more personalized and effective content delivery.

CCS Concepts

• **Information systems** → **Recommender systems**.

Keywords

Content-based Recommendation, LLM for RS, Library

ACM Reference Format:

Qijiong Liu, Lu Fan, and Xiao-Ming Wu. 2025. LEGOMMENDERS : A Comprehensive Content-Based Recommendation Library with LLM Support. In *Companion Proceedings of the ACM Web Conference 2025 (WWW Companion '25), April 28-May 2, 2025, Sydney, NSW, Australia.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3701716.3715305

1 Introduction

In online content discovery, recommender systems play a pivotal role as navigators, significantly enhancing user experiences through personalized content delivery. Traditionally, recommender systems have predominantly relied on transductive learning mechanisms [8, 28]. This approach utilizes static user and item identifiers (IDs) to generate predictions based on existing data. While effective within the confines of known datasets, this method presents limitations. It struggles to adapt to new users and items, often referred to as the "cold start" problem, and is less responsive to shifts in user preferences over time.

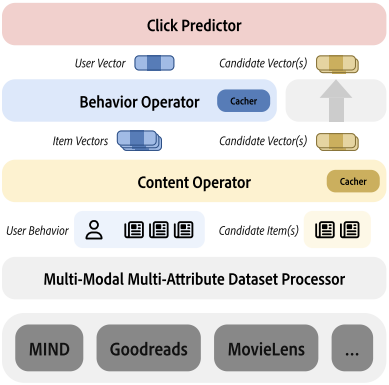

Figure 1: Overview of the LEGOMMENDERS package.

Modern recommender systems have shifted from transductive learning to inductive learning, utilizing inherent content features and user historical behaviors to create more dynamic models [7, 18, 20–22, 30, 32, 35, 36, 40]. Specifically, a content-based recommendation model typically consists of three components: (1) A *content operator* that generates embeddings for both the candidate item and each item in the user's behavior sequence. (2) A *behavior operator* that fuses the user sequence into a unified user embedding. (3) A *click predictor* that calculates the click probability for the user on the given item. Notably, the content operator can either be trained jointly with the other two modules or be decoupled from them.

However, most existing recommender system libraries [3, 38, 41] employ a decoupled design, which fails to adapt the content encoder to specific recommendation scenarios. Typically, item embeddings are generated by a *pretrained* content encoder and used as an initial step. Although this approach enhances model efficiency, it has a significant limitation: the pretrained embeddings are often too general and not well-aligned with the specific recommendation context, leading to suboptimal recommendations.

In contrast, our LEGOMMENDERS library offers a unique and innovative feature by enabling the joint training of content operators alongside other modules. This capability allows for the seamless integration of content understanding directly into the recommendation pipeline. As shown in Figure 1, LEGOMMENDERS comprises four core components: the dataset processor, content operator, behavior operator, and click predictor. By combining these built-in operators and predictors in the configuration files, researchers can design over **1,000** recommendation models, utilizing 15 content operators, 8 behavior operators, and 9 click predictors. Remarkably, **95%** of these

*Xiao-Ming Wu is the corresponding author.
[1]https://github.com/Legommenders/Legommenders

WWW Companion '25, Sydney, NSW, Australia

ACM ISBN 979-8-4007-1331-6/2025/04
https://doi.org/10.1145/3701716.3715305

Table 1: Comparison to exisiting recommendation benchmarks (✓ | – | × means totally | partially | not met, respectively). "Partially met" indicates incomplete availability.

Feature Year	TorchRec (2022)	DeepCTR (2017)	DeepRec (2022)	RecBole (2022)	FuxiCTR (2021)	BARS (2022)	Ducho (2024)	LEGOMMENDERS *(ours)*
Content-based	×	×	×	–	–	–	✓	✓
LLM Encoding	×	×	×	×	×	×	✓	✓
End-to-end Training	×	×	×	×	×	×	×	✓
Fast Evaluation	×	×	×	×	×	×	×	✓
# Models	<10	30+	10+	150+	50+	50+	<10	1000+

models have not been tested or published before. These models can be evaluated across multiple content-based recommendation scenarios using over **15** datasets. Moreover, LEGOMMENDERS is the *first* library to inherently support large language models (LLMs). It not only accepts augmented data generated by LLMs for training but also integrates open-source LLMs as content operators/encoders. This dual capability allows LEGOMMENDERS to improve data quality and generate superior data embeddings using LLMs.

LEGOMMENDERS is designed to offer researchers and practitioners a comprehensive, flexible, and user-friendly platform for conducting experiments and analyses in content-based recommendation in the era of LLMs, aiming to facilitate new research directions in the field. The LEGOMMENDERS library, including code, data, and documentation, is accessible at: https://github.com/Jyonn/Legommenders.

2 Comparison to Existing Benchmarks

Despite the success of previous research, there remains a significant lack of standardized benchmarks and uniform evaluation protocols for content-based recommendation systems. As summarized in Table 1, traditional recommendation libraries typically accept *only ID-based* features and do not utilize large language models (LLMs) for content encoding. The recent library Ducho [3] offers multimodal feature extraction for downstream recommendation models but maintains a decoupled design. In contrast, LEGOMMENDERS is currently the only library that supports end-to-end training of content operators, behavior operators, and click predictors. Furthermore, we have developed an inference caching pipeline that achieves up to a 50x speedup in evaluation. By enabling easy modular combinations, we provide over 1,000 models, which is six times more than the largest existing model libraries.

3 LEGOMMENDERS: Details and Usage

In this section, we first introduce the supported recommendation tasks, followed by a detailed discussion of the key components. We then present our caching pipeline and conclude with an overview of the algorithm flow.

3.1 Recommendation Tasks

The LEGOMMENDERS library supports two fundamental recommendation tasks: **matching** and **ranking**.

In the matching task, given a user and a set of $K+1$ candidate items (one positive and K negative), the model performs a $K+1$ classification task to identify the positive item, formulated as:

$$\hat{y}_{ui} = \text{softmax}(f(x_u, x_i)),$$

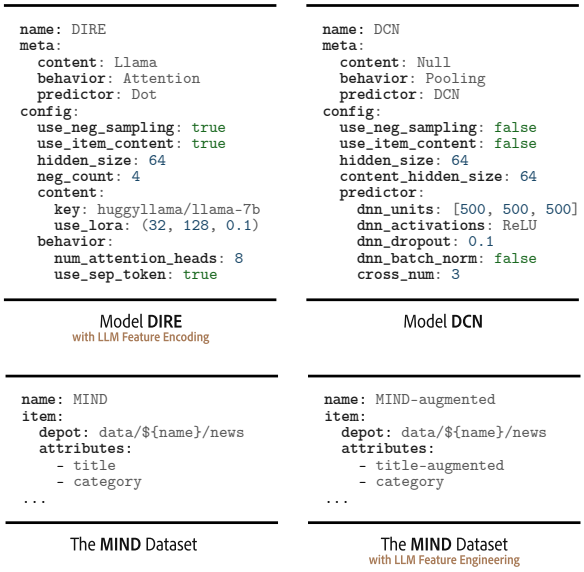

Figure 2: Examples for model and dataset configurations.

where $f(x_u, x_i)$ is mostly a simple dot operation, calculates the user-item feature relevance. The training objective is to maximize the likelihood of the correct positive item:

$$\mathcal{L}_{\text{matching}} = -\sum_{(u,i)\in\mathcal{D}} \sum_{i=1}^{K+1} y_{ui} \log(\hat{y}_{ui}),$$

where y_{ui} is the binary label for item i, and $\mathcal{D}$ is the dataset of user-item pairs.

In the ranking task, the model predicts the click probability for a given user-item pair, denoted as:

$$\hat{r}_{ui} = f(x_u, x_i),$$

where $f(x_u, x_i)$ can be deep CTR models and trained to minimize the mean squared error between the predicted and actual labels:

$$\mathcal{L}_{\text{ranking}} = \frac{1}{|\mathcal{D}|} \sum_{(u,i)\in\mathcal{D}} (y_{ui} - \hat{r}_{ui})^2.$$

The task can be configured with the `use_neg_sampling: false` and `neg_count: 0` parameters in the model configuration file, as depicted in Figure 2.

3.2 Dataset Processor

Legommenders provides a range of built-in processors for various recommendation scenarios, including news recommendation (e.g., MIND [34] and Eb-NerD [14]), book recommendation (e.g., Goodreads [27] and Amazon Books [10]), movie recommendation (e.g., Movielens [9] and Netflix [1]), and music recommendation (e.g., Amazon CDs [10] and Last.fm [23]). Dataset-specific processors convert the data into a unified format using the UniTok library[2].

In contrast to FuxiCTR [41] and BARS [39], which merge multiple tables into one, Legommenders adheres to the second normal form, significantly reducing data redundancy. This decoupled storage design allows Legommenders to easily accommodate augmented data generated by large language models simply by modifying the selected item attributes, as shown in Figure 2.

3.3 Content Operator

Legommenders decomposes content-based recommendation models into the content operator, behavior operator, and click predictor. This modular design allows for flexible selection and combination of these components to create new recommenders. The built-in content operators include average pooling used by text-based CTR models [8, 28], convolutional neural networks (CNN) [15] used by NAML [30] and LSTUR [2], Attention [26] used by NRMS [31] and PREC [19], and Fastformer [33].

Legommenders supports the use of LLMs as content operators, such as BERT [13], LLaMA [25], and other open-source models available on Hugging Face[3]. Building on insights from previous works [17, 19, 32], we propose a training method that freezes the lower layers while fine-tuning the upper layers, including LoRA-based PEFT [11]. This approach achieves up to 100x training acceleration compared to full fine-tuning, as it requires only the parameter "`--mode split --layer <N>`".

Additionally, Legommenders is compatible with identifier-based recommenders by setting `use_item_content: false` and using a randomly initialized item embedding table, such as DCN [28] model in Figure 2. It is compatible with decoupled content operator designs, like Ducho [3], by setting `use_item_content: false` and using an embedding table from pretrained models.

3.4 Behavior Operator and Click Predictor

The built-in behavior operators encompass several mechanisms, including average pooling, which is utilized by CTR models; Additive Attention [4], employed by NAML [30]; GRU [6], used by LSTUR [2]; and Attention, which is implemented in NRMS [31], BST [5], and PLM-NR [32]. Additionally, PolyAttention is utilized by MINER [16], among others. The built-in click predictors include the dot product, a method widely used in numerous matching-based models and various CTR models [8, 28, 29], which rely on feature interaction modules as their core design.

3.5 Caching Pipeline

During inference and evaluation, the model parameters are fixed. Traditional recommendation libraries and content-based recommender systems dynamically encode user and item embeddings for each user-item pair in the test set and calculate the click probability in real-time. This results in redundant computations, which can be exacerbated by the cascaded design of content and behavior operators. To mitigate this, we propose content and behavior cachers, as illustrated in Figure 1, which precompute and store embeddings for all items and users during the inference phase. During subsequent inferences, only the lightweight click predictor is required. The acceleration gained from caching becomes more pronounced as the frequency of repeated users and items, as well as with the sizes of the content and behavior operators. In some cases, this approach can yield up to 50x inference speedup. The caching mechanism is enabled by default and seamlessly integrated into the recommendation model, demonstrated in Algorithm 1.

Algorithm 1 Python-style Code for Training and Inference

```
class Legommenders:
    def forward(self, user, item, labels):
        content_op = self.content_cacher
        if self.content_op and self.training:
            content_op = self.content_op
        user = content_op(user)
        item = content_op(item)

        behavior_op = self.behavior_cacher
        if self.training:
            behavior_op = self.behavior_op
        user = behavior_op(user)

        scores = self.predictor(user, item)
        if self.training:
            return self.loss_fct(scores, labels)
        return scores
```

4 Experiments

In this section, we present a selection of benchmark results for representative models on the MIND dataset. These results illustrate the robust modular composition capabilities of Legommenders and its support for LLMs.

All baselines use the same hyperparameters, including embedding dimension, number of attention heads, learning rate, and others, to ensure consistency. Due to space limitations, we will provide the full configurations in our repository for reproducibility. The results show that: 1) models trained on GPT-augmented datasets consistently outperform those using the original datasets; 2) baselines incorporating more complex language models tend to achieve better performance; 3) LLM-finetuning scheme outperforms the decoupled design. These findings highlight the strong content understanding capabilities of LLMs, further underscoring the contribution of our library.

5 Conclusion

We have introduced Legommenders, a library designed for content-based recommendation systems, which stands out due to its ability to jointly train content operators, behavior operators, and click predictors for inductive learning, its modular design, and its support

[2]https://pypi.org/project/UniTok/
[3]https://huggingface.co

Table 2: Selected benchmark results on the MIND dataset. "ContentOp" and "BehaviorOp" denote Content Operator and Behavior Operator, respectively. "Original" and "Augmented" refer to the original MIND dataset and the GPT-augmented dataset as in ONCE [17]. Null(Llama1) indicates the decoupled design using Llama for item embedding extraction.

Dataset	Model	ContentOp	BehaviorOp	Predictor	AUC	MRR	N@5
Original	NAML_{ID}	Null	AdditiveAttention	Dot	50.13	23.01	22.35
Original	DCN	Null	Pooling	DCN	53.92	25.18	24.43
Original	DIN	Null	Null	DIN	55.95	25.88	25.95
Original	DCN_{text}	Pooling	Pooling	DCN	62.63	29.73	30.52
Original	DIN_{text}	Pooling	Null	DIN	62.90	30.06	30.65
Original	NAML	CNN	AdditiveAttention	Dot	61.75	30.60	31.35
Original	NRMS	Attention	Attention	Dot	61.71	30.20	30.98
Original	MINER	BERT	PolyAttention	Attention	63.88	32.19	33.04
Original	Fastformer	Fastformer	Fastformer	Dot	62.26	31.14	31.90
Original	PLM-NR	BERT	Attention	Dot	64.08	31.24	32.35
Original	DIRE	Llama1	Attention	Dot	68.50	36.21	38.11
Original	DIRE	Null(Llama1)	Attention	Dot	68.10	35.33	36.91
Augmented	DCN_{text}	Pooling	Pooling	DCN	65.77	32.86	34.10
Augmented	NAML	CNN	AdditiveAttention	Dot	63.88	32.17	33.14
Augmented	NRMS	Attention	Attention	Dot	63.71	32.14	33.11
Augmented	PLM-NR	BERT	Attention	Dot	65.13	32.98	34.30
Augmented	ONCE	Llama1	Attention	Dot	68.74	36.66	38.60

for LLMs as both content encoders and data generators. We believe that LEGOMMENDERS will serve as a valuable tool, significantly accelerating research within the recommendation community.

References

[1] Nicholas Ampazis. 2008. Collaborative filtering via concept decomposition on the netflix dataset. In *ECAI*, Vol. 8. 26–30.

[2] Mingxiao An, Fangzhao Wu, Chuhan Wu, Kun Zhang, Zheng Liu, and Xing Xie. 2019. Neural news recommendation with long-and short-term user representations. In *ACL*. 336–345.

[3] Matteo Attimonelli, Danilo Danese, Daniele Malitesta, Claudio Pomo, Giuseppe Gassi, and Tommaso Di Noia. 2024. Ducho 2.0: Towards a More Up-to-Date Unified Framework for the Extraction of Multimodal Features in Recommendation. In *WWW*. ACM.

[4] Dzmitry Bahdanau. 2014. Neural machine translation by jointly learning to align and translate. *arXiv* (2014).

[5] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior sequence transformer for e-commerce recommendation in alibaba. In *DLP-KDD*.

[6] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv* (2014).

[7] Junchen Fu, Xuri Ge, Xin Xin, Alexandros Karatzoglou, Ioannis Arapakis, Jie Wang, and Joemon M Jose. 2024. IISAN: Efficiently adapting multimodal representation for sequential recommendation with decoupled PEFT. In *SIGIR*. 687–697.

[8] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv* (2017).

[9] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *TIIS* 5, 4 (2015), 1–19.

[10] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *WWW*.

[11] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv* (2021).

[12] Dmytro Ivchenko, Dennis Van Der Staay, Colin Taylor, Xing Liu, Will Feng, Rahul Kindi, Anirudh Sudarshan, and Shahin Sefati. 2022. Torchrec: a pytorch domain library for recommendation systems. In *RecSys*. 482–483.

[13] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*. 4171–4186.

[14] Johannes Kruse, Kasper Lindskow, Saikishore Kalloori, Marco Polignano, Claudio Pomo, Abhishek Srivastava, Anshuk Uppal, Michael Riis Andersen, and Jes Frellsen. 2024. EB-NeRD a large-scale dataset for news recommendation. In *Proceedings of the Recommender Systems Challenge 2024*. 1–11.

[15] Yann LeCun, Bernhard Boser, John Denker, Donnie Henderson, Richard Howard, Wayne Hubbard, and Lawrence Jackel. 1989. Handwritten digit recognition with a back-propagation network. *NeurIPS* 2 (1989).

[16] Jian Li, Jieming Zhu, Qiwei Bi, Guohao Cai, Lifeng Shang, Zhenhua Dong, Xin Jiang, and Qun Liu. 2022. MINER: Multi-interest matching network for news recommendation. In *Findings of ACL*. 343–352.

[17] Qijiong Liu, Nuo Chen, Tetsuya Sakai, and Xiao-Ming Wu. 2024. Once: Boosting content-based recommendation with both open-and closed-source large language models. In *WSDM*. 452–461.

[18] Qijiong Liu, Hengchang Hu, Jiahao Wu, Jieming Zhu, Min-Yen Kan, and Xiao-Ming Wu. 2024. Discrete Semantic Tokenization for Deep CTR Prediction. In *WWW*. 919–922.

[19] Qijiong Liu, Jieming Zhu, Quanyu Dai, and Xiao-Ming Wu. 2022. Boosting deep CTR prediction with a plug-and-play pre-trainer for news recommendation. In *COLING*. 2823–2833.

[20] Qijiong Liu, Jieming Zhu, Quanyu Dai, and Xiao-Ming Wu. 2024. Benchmarking News Recommendation in the Era of Green AI. In *WWW*. 971–974.

[21] Qijiong Liu, Jieming Zhu, Lu Fan, Zhou Zhao, and Xiao-Ming Wu. 2024. STORE: Streamlining Semantic Tokenization and Generative Recommendation with A Single LLM. *arXiv* (2024).

[22] Qijiong Liu, Jieming Zhu, Yanting Yang, Quanyu Dai, Zhaocheng Du, Xiao-Ming Wu, Zhou Zhao, Rui Zhang, and Zhenhua Dong. 2024. Multimodal pretraining, adaptation, and generation for recommendation: A survey. In *SIGKDD*.

[23] Markus Schedl. 2016. The lfm-1b dataset for music retrieval and recommendation. In *ICMR*. 103–110.

[24] Weichen Shen. 2017. DeepCTR: Easy-to-use,Modular and Extendible package of deep-learning based CTR models. https://github.com/shenweichen/deepctr.

[25] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. LLaMA: open and efficient foundation language models. arXiv. *arXiv* (2023).

[26] A Vaswani. 2017. Attention is all you need. *NeurIPS* (2017).

[27] Mengting Wan and Julian J. McAuley. 2018. Item recommendation on monotonic behavior chains. In *RecSys*, Sole Pera, Michael D. Ekstrand, Xavier Amatriain, and John O'Donovan (Eds.). ACM, 86–94.

[28] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*.

[29] WM Wang, YH Peng, J Hu, and ZM Cao. 2009. Collaborative robust optimization under uncertainty based on generalized dynamic constraints network. *Structural and multidisciplinary optimization* 38 (2009), 159–170.

[30] Chuhan Wu, Fangzhao Wu, Mingxiao An, Jianqiang Huang, Yongfeng Huang, and Xing Xie. 2019. Neural News Recommendation with Attentive Multi-View Learning. In *IJCAI*. 3863–3869. https://doi.org/10.24963/ijcai.2019/536

[31] Chuhan Wu, Fangzhao Wu, Suyu Ge, Tao Qi, Yongfeng Huang, and Xing Xie. 2019. Neural news recommendation with multi-head self-attention. In *EMNLP-IJCNLP*.

[32] Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021. Empowering news recommendation with pre-trained language models. In *SIGIR*. 1652–1656.

[33] Chuhan Wu, Fangzhao Wu, Tao Qi, Yongfeng Huang, and Xing Xie. 2021. Fastformer: Additive attention can be all you need. *arXiv* (2021).

[34] Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, and Ming Zhou. 2020. MIND: A Large-scale Dataset for News Recommendation. In *ACL*. Association for Computational Linguistics.

[35] Chiyu Zhang, Yifei Sun, Jun Chen, Jie Lei, Muhammad Abdul-Mageed, Sinong Wang, Rong Jin, Sem Park, Ning Yao, and Bo Long. 2024. SPAR: Personalized Content-Based Recommendation via Long Engagement Attention. *arXiv* (2024).

[36] Chiyu Zhang, Yifei Sun, Minghao Wu, Jun Chen, Jie Lei, Muhammad Abdul-Mageed, Rong Jin, Angli Liu, Ji Zhu, Sem Park, et al. 2024. EmbSum: Leveraging the Summarization Capabilities of Large Language Models for Content-Based Recommendations. In *RecSys*. 1010–1015.

[37] Shuai Zhang, Yi Tay, Lina Yao, Aixin Sun, and Ce Zhang. 2022. *Deep Learning for Recommender Systems*. Springer US, New York, NY, 173–210.

[38] 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. 2022. RecBole 2.0: Towards a More Up-to-Date Recommendation Library. In *CIKM*. ACM, 4722–4726.

[39] Jieming Zhu, Quanyu Dai, Liangcai Su, Rong Ma, Jinyang Liu, Guohao Cai, Xi Xiao, and Rui Zhang. 2022. BARS: Towards Open Benchmarking for Recommender Systems. In *SIGIR*. ACM, 2912–2923.

[40] Jieming Zhu, Mengqun Jin, Qijiong Liu, Zexuan Qiu, Zhenhua Dong, and Xiu Li. 2024. CoST: Contrastive Quantization based Semantic Tokenization for Generative Recommendation. In *RecSys*. 969–974.

[41] Jieming Zhu, Jinyang Liu, Shuai Yang, Qi Zhang, and Xiuqiang He. 2021. Open Benchmarking for Click-Through Rate Prediction. In *CIKM*. ACM, 2759–2769.