# Non-Recursive Cluster-Scale Graph Interacted Model for Click-Through Rate Prediction

Yuanchen Bei*
Zhejiang University
Hangzhou, China
yuanchenbei@zju.edu.cn

Hao Chen*
The Hong Kong Polytechnic University
Hung Hom, Hong Kong SAR
sundaychenhao@gmail.com

Shengyuan Chen
The Hong Kong Polytechnic University
Hung Hom, Hong Kong SAR
shengyuan.chen@connect.polyu.hk

Xiao Huang
The Hong Kong Polytechnic University
Hung Hom, Hong Kong SAR
xiaohuang@comp.polyu.edu.hk

Sheng Zhou†
Zhejiang University
Hangzhou, China
zhousheng_zju@zju.edu.cn

Feiran Huang†
Jinan University
Guangzhou, China
huangfr@jnu.edu.cn

## ABSTRACT

Extracting users' interests from their behavior, particularly their 1-hop neighbors, has been shown to enhance Click-Through Rate (CTR) prediction performance. However, online recommender systems impose strict constraints on the inference time of CTR models, which necessitates pruning or filtering users' 1-hop neighbors to reduce computational complexity. Furthermore, while the graph information of users and items has been proven effective in collaborative filtering models, recursive graph convolution can be computationally costly and expensive to implement. To address these challenges, we propose the Non-Recursive Cluster-scale Graph Interacted (NRCGI) model, which reorganizes graph convolutional networks in a non-recursive and cluster-scale view to enable CTR models to consider deep graph information with low computational cost. NRCGI employs non-recursive cluster-scale graph aggregation, which allows the online recommendation computational complexity to shrink from tens of thousands of items to tens to hundreds of clusters. Additionally, since NRCGI aggregates neighbors in a non-recursive view, each hop of neighbors has a clear physical meaning. NRCGI explicitly constructs meaningful interactions between the hops of neighbors of users and items to fully model users' intent towards the given item. Experimental results demonstrate that NRCGI outperforms state-of-the-art baselines in three public datasets and one industrial dataset while maintaining efficient inference.

## CCS CONCEPTS

• **Information systems → Computational advertising**; • **Human-centered computing → Social recommendation**.

*Both authors contributed equally to this research.
†Corresponding author.


*CIKM '23, October 21–25, 2023, Birmingham, United Kingdom.*

ACM ISBN 979-8-4007-0124-5/23/10…$15.00
https://doi.org/10.1145/3583780.3615180

## KEYWORDS

graph-based CTR prediction, online efficiency, graph interaction

**ACM Reference Format:**
Yuanchen Bei, Hao Chen, Shengyuan Chen, Xiao Huang, Sheng Zhou, and Feiran Huang. 2023. Non-Recursive Cluster-Scale Graph Interacted Model for Click-Through Rate Prediction. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3583780.3615180

## 1 INTRODUCTION

Click-through rate (CTR) prediction is a vital task in recommender systems, especially in online environments where user behavior data offers valuable insights into their interests [3, 14, 20, 21, 37]. As a result, interest-based CTR models that learn user intent from their historical behaviors have achieved state-of-the-art performance in CTR prediction [30, 39, 41, 42]. Specifically, these models treat the historical behaviors of a target user as item sequences and employ either transformers [31, 33] or sequential-based structures [23, 42] to extract the user's intents towards a target item. However, the computational complexity of these models grows linearly with the length of input sequences. Given the strict constraints on online inference speed, it is crucial to reduce the length of input sequences while ensuring the sequences contain sufficient information.

To balance the trade-off between inference speed and recommendation performance, some CTR models truncate user historical behaviors [13, 38, 41, 42]. For instance, DIN [42] truncates the most recent interactions and utilizes a local activation unit to adaptively learn the representation of user interests from historical behaviors based on the attention mechanism. Similarly, DIEN [41] models the evolving interest with attention-enhanced GRU [7]. Another line of research aims to filter out unrelated items [25, 26, 28, 32]. For example, SIM [26] proposes two cascaded search units to extract user interests based on the searched relevant items.

Despite the success of existing models in learning from user behavior data, there remains a gap in the development of effective structures for reducing the order of magnitude of the input sequence. Recent research has shown that Graph Neural Network (GNN)-based models, such as NGCF [34] and LightGCN [19], have achieved

significant improvements in predicting user-item interactions by leveraging deep graph information [2, 9, 15]. However, these GNN-based models rely on recursive item-level graph convolution, which can lead to an exponential increase in computational cost [4, 5, 11]. While effective, incorporating item-level graph convolution in online CTR prediction models can significantly impact online inference time and adversely affect the user experience. Therefore, it is crucial to develop an efficient structure that can reduce the order of magnitude of the input sequence to handle the exponential growth of neighbors without compromising inference speed.

To address the aforementioned problem, we propose a novel Non-Recursive Cluster-scale Graph Interacted (NRCGI) CTR model. Unlike recursive item-level graph convolutional models, the NRCGI model proposes a non-recursive cluster-scale graph convolutional solution. By using non-recursive graph convolutional techniques, we can pre-compute the graph representation of each user and item, which accelerates the online inference process. We propose a bi-parted graph classification strategy that groups users with similar behaviors into the same cluster and does the same for items. Since users or items with similar behaviors have similar embeddings, conducting non-recursive graph convolutions on a cluster scale can reduce the order of magnitudes from item level to cluster level, namely, from tens of thousands to tens to hundreds. Moreover, the aggregated clusters can better describe a user's intent than providing a mass of items. Furthermore, non-recursive cluster-scale graph convolutions preserve the physical meanings of each hop of neighbors, for instance, a user's 1-hop neighbors represent their behavior histories, and their 2-hop neighbors represent users with similar interests. Based on this, we propose a graph interaction model that comprehensively considers all possible interactions between the hop of neighbors of users and items. These approaches allow NRCGI to fully exploit graph information to enhance CTR prediction while maintaining efficient online inference. In summary, our contribution can be concluded as follows:

- NRCGI utilizes Non-Recursive Cluster-scale Extraction (NRCE) to efficiently and meaningfully pre-aggregate the neighbors of users and items offline. With the assistance of Graph-Interacted Prediction (GIP), NRCGI enhances CTR prediction using graph information while maintaining efficient online inference.
- We propose the global graph clustering to cluster users and items based on their behaviors, and then we introduce offline cluster-scale plain-aggregation to achieve non-recursive cluster-scale information extraction.
- We propose the online GIP structure to aggregate the cluster-scale embeddings and explicitly construct meaningful graph information interactions, comprehensively considering the implicit relations between the graph structures of users and items.
- Extensive experiments on three public datasets and one industrial dataset demonstrate that NRCGI outperforms state-of-the-art CTR models in terms of both efficiency and effectiveness.[1]

## 2 METHODOLOGY

As depicted in Figure 1, in the Non-Recursive Cluster-scale Extraction (NRCE) stage, NRCGI pre-aggregates individual nodes into clusters, and then into several embedding vectors for any given hop of neighbors, reducing their number by several magnitudes compared to the original neighbors. The Graph-Interacted Prediction (GIP) stage then aggregates the cluster embeddings and constructs the graph interaction to predict the click-through rate. In this section, we elaborate on each component of our approach.

### 2.1 Non-Recursive Cluster-scale Extraction

As shown in the offline part of Figure 1, NRCE first globally clusters all users and items into several clusters. Thus, given any hop of neighbors, NRCE directly assigns the neighbors to their corresponding clusters. On one hand, global clustering can easily cluster any given hop of neighbors. On the other hand, global clustering can model users' interests in more macroscopic views, thus will be more robust and less affected by individual users.

**Global Graph Clustering.** As shown in the bottom of Figure 1, we inspire by and modify the Louvain algorithm [12, 22] to co-cluster the users and the items on the global user-item bipartite graph. The core idea is to define the graph modularity $Q$ in the user-item interaction graph and maximize it iteratively [8, 40]. $Q$ is defined as:

$$Q = \frac{1}{m} \sum_{c \in C} (\psi_c^{in} - \frac{\psi_c^{user} \cdot \psi_c^{item}}{m}), \tag{1}$$

where $m$ is the total number of edges, $C$ is the set of clusters, $\psi_c^{in}$ is the number of edges inside cluster $c$, and $\psi_c^{user}$ and $\psi_c^{item}$ represent the sum of degrees of all users and all items in cluster $c$, respectively.

Within each iteration, the goal is to maximize the modularity gain $\Delta Q$. Formally, for the specific user node $a$, its modularity gain is defined as follows:

$$\Delta Q = \frac{k_{a,c}^{item}}{m} - \frac{k_a \cdot \psi_c^{item}}{m^2}, \tag{2}$$

where $k_{a,c}^{item}$ is the number of all edges between node $a$ and all item nodes in the cluster $c$, $k_a$ is the degree of node $a$. The modularity gain for item nodes can be calculated in a similar way. Finally, after the modularity converges, we have several clusters that cluster the users and items together according to the global interactions. Then by splitting the users and items, we have the user clusters and the item clusters, respectively.

**Cluster-scale Plain-Aggregation.** For each user/item, NRCE first extracts its graph structure by hop-order traversal, unfolding its 3-layer subgraph into two additional node lists, representing its 1-hop and 2-hop neighbors, respectively. For a given hop $l$ and node $v$, the sampled $l$-hop neighbors of node $v$ can be formally represented with $\mathcal{N}_v^{(l)} = \{n_{v,1}^{(l)}, \cdots, n_{v,|\mathcal{N}_v^{(l)}|}^{(l)}\}$, where $|\mathcal{N}_v^{(l)}|$ denotes the number of the $l$-hop neighbors for node $v$. By assigning each node with its corresponding cluster, the given hop of neighbors can be present as a set of clusters, namely $\mathcal{C}_v^{(l)}$. Then, the cluster-wise embeddings can be pre-aggregated as:

$$\boldsymbol{e}_{v,c}^{(l)} = Agg\{\boldsymbol{e}_j, \ for \ j \in \ \boldsymbol{c}_{v,c}^{(l)}\}, \tag{3}$$

where $Agg$ denotes the aggregation function of the cluster, which we select sum pooling by default.

After the offline NRCE stage, we store the pre-aggregated cluster-wise embeddings of the sampled 1-hop and 2-hop neighbors of each user/item into memories or key-value databases [17, 27] to be further utilized in the following online GIP stage.

[1] Source code is available at https://github.com/YuanchenBei/NRCGI.

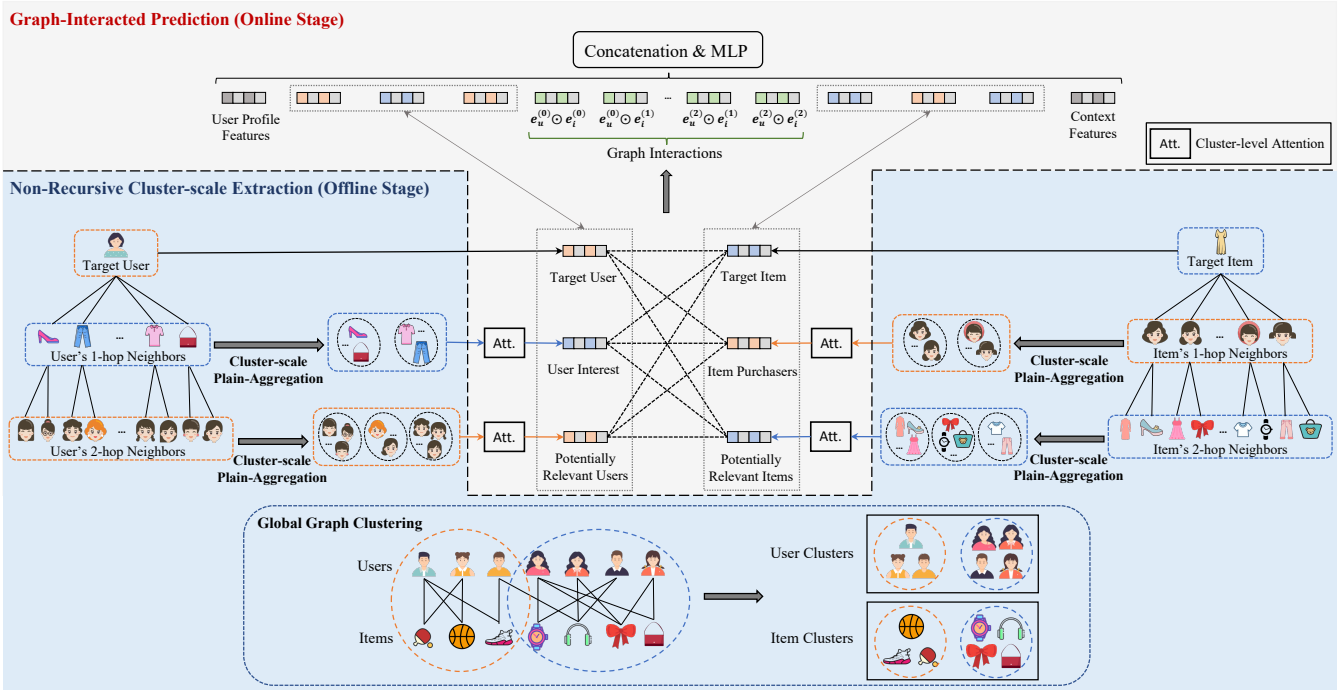


**Figure 1: The overall framework of NRCGI, consists of two stages: (i) the offline Non-Recursive Cluster-scale Extraction (NRCE) stage: we first globally cluster users and items using a tailored clustering method. Then, we assign multi-hop neighbors into clusters based on global graph clustering and pre-aggregate the cluster-level embeddings; (ii) the online Graph-Interacted Prediction (GIP) stage: we aggregate the cluster-level embeddings and construct graph interaction terms to predict the CTR.**

## 2.2 Graph-Interacted Prediction

GIP first obtains hop-wise representations of the target user/item and then mines their relations with the graph-interacted structures. Unlike existing GNN models [19, 34], which merge all graph embeddings into one super vector [1, 36], GIP explicitly specifies the meanings of each hop of neighbors and constructs meaningful graph interactions for better modeling the user-item relations.

As presented as the online part in Figure 1, GIP first obtains hop-wise representations of the target user/item by cluster-level attention mechanism. Then for each cluster $c$ in $C_v^{(l)}$, its attention weights can be computed as:

$$\alpha_{v,c}^{(l)} = Tanh(\boldsymbol{W}_2 \cdot ((\boldsymbol{W}_1 \cdot (\boldsymbol{e}_v^{(0)} \odot \boldsymbol{e}_{v,c}^{(l)})) + b), \tag{4}$$

where $\odot$ is the Hadamard product operation, and $\boldsymbol{W}_1$, $\boldsymbol{W}_2$, and $b$ denote the trainable parameters of the attention mechanism. By computing the attention weights, the representation of the $l$-hop neighbors can be given as:

$$\boldsymbol{e}_v^{(l)} = \sum_{c \in C_v^{(l)}} \alpha_{v,c}^{(l)} \cdot \boldsymbol{e}_{v,c}^{(l)}. \tag{5}$$

For convenience, we utilize $\boldsymbol{e}_v^{(0)}$ to denote the own embedding of the given node $v$.

Since the above process is a plain neighbor aggregation method, the aggregated embeddings of different hops of neighbors maintain different physical meanings in the user-item bipartite graph. To be specific, for user $u$, the user's 1-hop graph embedding $\boldsymbol{e}_u^{(1)}$ represents the user's interest, while the user's 2-hop graph embedding $\boldsymbol{e}_u^{(2)}$ contains the users which share similar interest with the user $u$, namely the potentially relevant users. Thus considering the relation between $\boldsymbol{e}_u^{(1)}$ and $\boldsymbol{e}_i^{(0)}$, it reflects whether the given item $i$ shares similar characters with the user's historical interest. Meanwhile, the relation between $\boldsymbol{e}_u^{(2)}$ and $\boldsymbol{e}_i^{(0)}$ can infer the intent of the potential related users of user $u$ on the given item $i$.

To explicitly extract all potential meaningful interactions, GIP computes the Cartesian product across the user's graph embeddings and the item's graph embeddings. Thus the inference function can be formally written as follows:

$$\hat{y} = MLP\left((\|_{l_u}^{L} \boldsymbol{e}_u^{(l_u)}) \parallel (\|_{l_i}^{L} \boldsymbol{e}_i^{(l_i)}) \parallel (\|_{l_u,l_i}^{L,L} \boldsymbol{e}_u^{(l_u)} \odot \boldsymbol{e}_i^{(l_i)})\right), \tag{6}$$

where $\parallel$ denotes the concatenating operation. Here, $L$ denotes the depth of the user/item sub-graphs, while $l_u$ and $l_i$ range for 0 to $L$.

# 3 EXPERIMENT

## 3.1 Experimental Setup

***Datasets***. We conduct experiments on three public datasets and an industrial dataset for model evaluation. Three public datasets include **MovieLens** dataset [18] (containing 10 million samples), **Electronics** subset (containing 1.7 million samples) and **Book** subset (containing 8.9 million samples) of Amazon dataset [24]. The industrial dataset is collected from the 14 days of video play logs on the Channels platform of **Weixin**, containing about 23 million anonymous samples. The total cluster numbers of MovieLens, Electronics, Book, and Weixin are 5, 93, 146, and 295, respectively. The dataset split follows the settings in previous works [28, 41].

**Table 1: Overall comparison results (↑: the higher, the better; ↓: the lower, the better). The best and second-best results are highlighted in bold font and underlined. * indicates the statistical significance $p < 0.01$ compared to the best-performed baseline.**

| Method | MovieLens | | | Electronics | | | Book | | | WeiXin | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AUC (↑) | GAUC (↑) | LogLoss (↓) | AUC (↑) | GAUC (↑) | LogLoss (↓) | AUC (↑) | GAUC (↑) | LogLoss (↓) | AUC (↑) | GAUC (↑) | LogLoss (↓) |
| PNN | 0.6778 | 0.6877 | 0.6430 | 0.7681 | 0.7718 | 0.5678 | 0.7394 | 0.7423 | 0.5847 | 0.7778 | 0.7791 | 0.5730 |
| Wide&Deep | 0.6827 | 0.6901 | 0.6387 | 0.7730 | 0.7747 | 0.5611 | 0.7444 | 0.7451 | 0.5798 | 0.7771 | 0.7788 | 0.5736 |
| DeepFM | 0.6793 | 0.6871 | 0.6412 | 0.7716 | 0.7742 | 0.5629 | 0.7428 | 0.7438 | 0.5825 | 0.7772 | 0.7799 | 0.5742 |
| AFM | 0.6878 | 0.6917 | 0.6348 | 0.7631 | 0.7628 | 0.5636 | 0.7270 | 0.7280 | 0.5913 | 0.7753 | 0.7718 | 0.6126 |
| DIN | 0.7178 | 0.7114 | 0.6147 | 0.8071 | 0.8080 | 0.5327 | 0.8212 | 0.8076 | 0.5171 | 0.8027 | 0.7896 | 0.5442 |
| DIEN | 0.7431 | 0.7320 | 0.6054 | 0.8204 | 0.8183 | 0.5221 | 0.8350 | 0.8172 | 0.5001 | 0.8035 | 0.7856 | 0.5409 |
| UBR4CTR | 0.6977 | 0.6965 | 0.6295 | 0.7767 | 0.7766 | 0.5534 | 0.7918 | 0.7855 | 0.5470 | 0.7912 | 0.7838 | 0.5601 |
| SIM | 0.7370 | 0.7278 | 0.6079 | 0.8213 | 0.8198 | 0.5179 | 0.8341 | 0.8169 | 0.5004 | 0.8050 | 0.7879 | 0.5560 |
| NGCF | 0.7004 | 0.6994 | 0.6259 | 0.8022 | 0.7999 | 0.5382 | 0.8537 | 0.8411 | 0.4743 | 0.8031 | 0.8007 | 0.5397 |
| LightGCN | 0.7049 | 0.7036 | 0.6223 | 0.8128 | 0.8109 | 0.5271 | 0.8731 | 0.8634 | 0.4470 | 0.8073 | 0.8051 | 0.5351 |
| NRCGI (ours) | **0.7595*** | **0.7506*** | **0.5808*** | **0.8411*** | **0.8410*** | **0.4956*** | **0.9025*** | **0.8953*** | **0.4001*** | **0.8314*** | **0.8176*** | **0.5045*** |
| Improv.% | 2.21% | 2.54% | 4.06% | 2.41% | 2.59% | 4.31% | 3.37% | 3.69% | 10.49% | 2.99% | 1.55% | 5.72% |

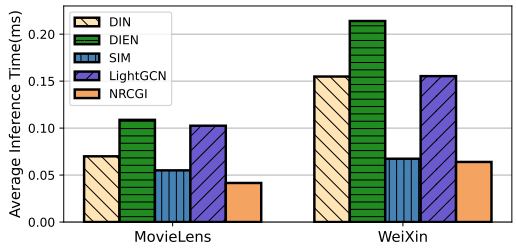


**Figure 2: Average online inference time comparison.**

***Baselines and metrics.*** We compare NRCGI with ten representative CTR prediction models, including feature-interaction models: PNN [29], Wide&Deep [6], DeepFM [16] and AFM [35]; user-interest models: DIN [42], DIEN [41], UBR4CTR [28] and SIM [26] (hard-search is adopted); and graph-based models: NGCF [34] and LightGCN [19]. To evaluate the performance, we adopt three widely used CTR prediction evaluation metrics, i.e., AUC, GAUC, and LogLoss [10, 42, 43]. Note that we run all the experiments *five* times with different random seeds and report the average results.

## 3.2 Main Results

Table 1 reports the performance comparison between NRCGI and the baseline models. The improvement is calculated by comparing NRCGI with the best baseline (underlined). From the results, we observe that, in general, our NRCGI outperforms all the other state-of-the-art CTR prediction models on all datasets with significant margins. Specifically, for the AUC metric, NRCGI outperforms the best baseline SIM by 2.21%, 2.41%, 3.37%, and 2.99% on MovieLens, Amazon-Electronics, Amazon-Book, and Weixin, respectively. These results verify that considering the graph information with NRCE and constructing the graph interactions with GIP help the CTR model to achieve better prediction performance.

## 3.3 Efficiency Study

Since CTR models need to infer the user's intent online, we compare the online response time per user-item pair between NRCGI and the top-4 performed baselines to present the inference efficiency. From Figure 2, we find that NRCGI achieves relatively the fastest inference. There are two main reasons: 1) The cluster-scale aggregation helps in decreasing the number of keys of the attention mechanism from hundreds of users/items to tens of clusters, reducing the computational complexity by magnitudes without filtering; 2) The cluster embeddings can be pre-computed and stored asynchronously every minute. Then, the online inference only focuses on aggregating the cluster-level embedding and thus reducing the inference time. Noted that we present the online serving inference time for SIM and NRCGI, where the item filtering results (SIM) and cluster embeddings (NGCGI) are computed asynchronously.

**Table 2: Ablation study results on AUC metric.**

| Variant | MovieLens | Electronics | Book | WeiXin |
|---|---|---|---|---|
| NRCGI | **0.7595** | **0.8411** | **0.9025** | **0.8314** |
| NRCGI-*w/o* NRCE | 0.7260 | 0.8293 | 0.9003 | 0.8204 |
| NRCGI-*w/o* GIP | 0.7487 | 0.8358 | 0.8958 | 0.8211 |

## 3.4 Ablation Study

We conduct the ablation study by comparing NRCGI with its two variations: "NRCGI-*w/o* NRCE" utilizes traditional recursive graph aggregation to extract the graph embeddings, while "NRCGI-*w/o* GIP" skips the graph interaction terms in the MLP function. According to Table 2, NRCGI significantly outperforms other variations. This verifies that NRCE is more powerful than recursive GNN models in CTR prediction and constructing graph interactions with GIP helps in better modeling the user-item relations.

# 4 CONCLUSION

In this paper, we propose a novel Non-Recursive Cluster-scale Graph Interacted model (NRCGI) for CTR prediction, which could efficiently extract and effectively exploit the graph embeddings to enhance CTR prediction performance. Specifically, by utilizing NRCE, NRCGI is able to extract graph embeddings for users/items without adding computational complexity. Meanwhile, GIP is further presented to be a powerful graph feature interaction structure. Extensive experiments demonstrate the effectiveness of NRCGI.

# 5 ACKNOWLEDGEMENT

This work is supported in part by the National Natural Science Foundation of China (Grant No. 61932010, U22A2095, 62272200, 62106221, 61972349), Zhejiang Provincial Natural Science Foundation of China (Grant No. LTGG23F030005), and Ningbo Natural Science Foundation (Grant No. 2022J183).

# REFERENCES


[1] Hao Chen, Zengde Deng, Yue Xu, and Zhoujun Li. 2021. Non-recursive graph convolutional networks. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 3205–3209.

[2] Hao Chen, Zhong Huang, Yue Xu, Zengde Deng, Feiran Huang, Peng He, and Zhoujun Li. 2022. Neighbor enhanced graph convolutional networks for node classification and recommendation. *Knowledge-Based Systems* 246 (2022), 108594.

[3] Hao Chen, Zefan Wang, Feiran Huang, Xiao Huang, Yue Xu, Yishi Lin, Peng He, and Zhoujun Li. 2022. Generative adversarial framework for cold-start item recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2565–2571.

[4] Hao Chen, Yue Xu, Feiran Huang, Zengde Deng, Wenbing Huang, Senzhang Wang, Peng He, and Zhoujun Li. 2020. Label-aware graph convolutional networks. In *Proceedings of the 29th ACM international conference on information & knowledge management*. 1977–1980.

[5] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2023. Bias and debias in recommender system: A survey and future directions. *ACM Transactions on Information Systems* 41, 3 (2023), 1–39.

[6] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.

[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Deep Learning and Representation Learning Workshop*.

[8] Pasquale De Meo, Emilio Ferrara, Giacomo Fiumara, and Alessandro Provetti. 2011. Generalized louvain method for community detection in large networks. In *2011 11th international conference on intelligent systems design and applications*. IEEE, 88–93.

[9] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph neural networks for social recommendation. In *The world wide web conference*. 417–426.

[10] Tom Fawcett. 2006. An introduction to ROC analysis. *Pattern recognition letters* 27, 8 (2006), 861–874.

[11] Chao Feng, Defu Lian, Xiting Wang, Zheng Liu, Xing Xie, and Enhong Chen. 2023. Reinforcement routing on proximity graph for efficient recommendation. *ACM Transactions on Information Systems* 41, 1 (2023), 1–27.

[12] Liang Feng, Qianchuan Zhao, and Cangqi Zhou. 2020. Improving performances of Top-N recommendations with co-clustering method. *Expert Systems with Applications* 143 (2020), 113078.

[13] Yufei Feng, Fuyu Lv, Weichen Shen, Menghan Wang, Fei Sun, Yu Zhu, and Keping Yang. 2019. Deep session interest network for click-through rate prediction. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. 2301–2307.

[14] Chongming Gao, Kexin Huang, Jiawei Chen, Yuan Zhang, Biao Li, Peng Jiang, Shiqi Wang, Zhong Zhang, and Xiangnan He. 2023. Alleviating Matthew Effect of Offline Reinforcement Learning in Interactive Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 238–248.

[15] Chen Gao, Yu Zheng, Nian Li, Yinfeng Li, Yingrong Qin, Jinghua Piao, Yuhan Quan, Jianxin Chang, Depeng Jin, Xiangnan He, et al. 2023. A survey of graph neural networks for recommender systems: Challenges, methods, and directions. *ACM Transactions on Recommender Systems* 1, 1 (2023), 1–51.

[16] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 1725–1731.

[17] Jing Han, Ee Haihong, Guan Le, and Jian Du. 2011. Survey on NoSQL database. In *2011 6th international conference on pervasive computing and applications*. IEEE, 363–366.

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

[19] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 639–648.

[20] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*. 173–182.

[21] Feiran Huang, Zefan Wang, Xiao Huang, Yufeng Qian, Zhetao Li, and Hao Chen. 2023. Aligning Distillation For Cold-Start Item Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1147–1157.

[22] Andrea Lancichinetti and Santo Fortunato. 2009. Community detection algorithms: a comparative analysis. *Physical review E* 80, 5 (2009), 056117.

[23] Pan Li, Zhichao Jiang, Maofei Que, Yao Hu, and Alexander Tuzhilin. 2021. Dual attentive sequential learning for cross-domain click-through rate prediction. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*. 3172–3180.

[24] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. 43–52.

[25] Qi Pi, Weijie Bian, Guorui Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Practice on long sequential user behavior modeling for click-through rate prediction. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2671–2679.

[26] Qi Pi, Guorui Zhou, Yujing Zhang, Zhe Wang, Lejian Ren, Ying Fan, Xiaoqiang Zhu, and Kun Gai. 2020. Search-based user interest modeling with lifelong sequential behavior data for click-through rate prediction. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2685–2692.

[27] Wittawat Puangsaijai and Sutheera Puntheeranurak. 2017. A comparative study of relational database and key-value database for big data applications. In *2017 International Electrical Engineering Congress (iEECON)*. IEEE, 1–4.

[28] Jiarui Qin, Weinan Zhang, Xin Wu, Jiarui Jin, Yuchen Fang, and Yong Yu. 2020. User behavior retrieval for click-through rate prediction. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2347–2356.

[29] Yanru Qu, Han Cai, Kan Ren, Weinan Zhang, Yong Yu, Ying Wen, and Jun Wang. 2016. Product-based neural networks for user response prediction. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 1149–1154.

[30] Qijie Shen, Hong Wen, Wanjie Tao, Jing Zhang, Fuyu Lv, Zulong Chen, and Zhao Li. 2022. Deep interest highlight network for click-through rate prediction in trigger-induced recommendation. In *Proceedings of the ACM Web Conference 2022*. 422–430.

[31] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 1441–1450.

[32] Huinan Sun, Guangliang Yu, Pengye Zhang, Bo Zhang, Xingxing Wang, and Dong Wang. 2022. Graph Based Long-Term And Short-Term Interest Model for Click-Through Rate Prediction. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 1818–1826.

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[34] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*. 165–174.

[35] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional factorization machines: learning the weight of feature interactions via attention networks. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 3119–3125.

[36] Yue Xu, Hao Chen, Zengde Deng, Yuanchen Bei, and Feiran Huang. 2022. Flattened Graph Convolutional Networks For Recommendation. In *4th Workshop on Deep Learning Practice and Theory for High-Dimensional Sparse and Imbalanced Data with KDD*.

[37] Yue Xu, Hao Chen, Zefan Wang, Jianwen Yin, Qijie Shen, Dimin Wang, Feiran Huang, Lixiang Lai, Tao Zhuang, Junfeng Ge, et al. 2023. Multi-factor Sequential Re-ranking with Perception-Aware Diversification. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 5327–5337.

[38] Zeping Yu, Jianxun Lian, Ahmad Mahmoody, Gongshen Liu, and Xing Xie. 2019. Adaptive User Modeling with Long and Short-Term Preferences for Personalized Recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. 4213–4219.

[39] Weinan Zhang, Jiarui Qin, Wei Guo, Ruiming Tang, and Xiuqiang He. 2021. Deep Learning for Click-Through Rate Estimation. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. 4695–4703.

[40] Cangqi Zhou, Liang Feng, and Qianchuan Zhao. 2018. A novel community detection method in bipartite networks. *Physica A: Statistical Mechanics and its Applications* 492 (2018), 1679–1693.

[41] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 5941–5948.

[42] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1059–1068.

[43] Jieming Zhu, Jinyang Liu, Shuai Yang, Qi Zhang, and Xiuqiang He. 2021. Open benchmarking for click-through rate prediction. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2759–2769.