同行专家业内评价意见书编号: 20240854210

附件1

浙江工程师学院（浙江大学工程师学院）同行专家业内评价意见书

姓名： 吴怡萱

学号： 22160173

申报工程师职称专业类别（领域）： 电子信息

浙江工程师学院（浙江大学工程师学院）制

2024年03月28日

一、个人申报

（一）基本情况【围绕《浙江工程师学院（浙江大学工程师学院）工程类专业学位研究生工程师职称评审参考指标》，结合该专业类别(领域)工程师职称评审相关标准，举例说明】

1.对本专业基础理论知识和专业技术知识掌握情况
本人对专业基础理论知识掌握扎实，在日常学习中，严谨自律，勤奋好学，专业成绩突出，于研一学年高质量地完成了所有课程，平均绩点92.36，位列班级前茅。在13门课程中，其中5门课程成绩在95分以上，10门课程成绩在90分以上。
2.工程实践的经历
本人在浙江数服智联科技有限公司进行专业实践，主要工作内容是利用大规模复杂图优化嵌入表示以提升下游任务性能，研究内容涉及图表示学习、图神经网络、对比学习。技术难点是用更轻量的图构造方法和更高效的聚合方法来提升图学习性能。
3.在实际工作中综合运用所学知识解决复杂工程问题的案例
本人系统地掌握了图嵌入学习和图对比学习的领域知识。图嵌入(Graph Embedding)是一种将图结构中的节点或边转化为低维向量表示的技术。近年来，随着深度学习和图神经网络的兴起，图嵌入得到了广泛的研究和应用。早期的图嵌入方法主要基于特征工程和矩阵分解等传统机器学习技术。这些方法使用节点的局部邻域信息或全局网络结构信息来构建节点之间的相似性或关联性，并通过降维算法将节点映射到低维空间。然而，这些方法在处理大规模复杂图时存在着效率和可扩展性的问题。为了克服这些限制，基于深度学习的图嵌入方法应运而生。这些方法利用神经网络模型，通过学习节点之间的局部结构和全局特征，将图中的节点映射到低维向量空间。深度学习方法的优势在于能够自动地从数据中学习到更丰富、更有表达力的特征表示，并且能够适应不同类型的图结构。随着研究的深入，出现了众多的图嵌入方法。其中，基于图卷积网络(Graph Convolutional Network，GCN)的方法成为了研究的热点。GCN通过在图上进行卷积操作，利用节点的邻居信息来更新节点的表示，从而捕捉节点之间的关系和上下文信息。此外，还有基于自编码器(Autoencoder)、生成对抗网络(Generative Adversarial Network，GAN)和变分自编码器(Variational Autoencoder，VAE)等方法，它们通过重构图结构或生成虚拟图来学习图的嵌入表示。图对比学习(Graph Contrastive Learning)是一种用于学习图表示技术，旨在通过对比不同图之间的相似性和差异性来训练图嵌入模型。它在最近几年内得到了广泛的关注和研究。图对比学习的发展可以追溯到传统的对比学习方法，如孪生网络和三元组损失。这些方法主要应用于图像和文本领域，通过比较不同样本之间的相似性来学习特征表示。然而，将对比学习方法推广到图结构数据上面临着更大的挑战，因为图数据的结构和属性更加复杂。随着图神经网络的发展，基于深度学习的图对比学习方法逐渐兴起。这些方法通常采用自监督学习的思想，通过设计合适的对比任务和损失函数，来学习图数据的判别性特征表示。其中，最常见的方法是构建图对比任务，如正负样本对比、同质性对比和异质性对比等。本次实践给我提供了实际应用和操作的机会，能够将学到的理论知识转化为实际技能。通过实践，我可以更深入地理解和掌握所学知识，并在实际中培养出更高水平的专业能力。
本次专业实践活动与学位论文选题紧密衔接，旨在通过对图嵌入的深入研究来为技术应用的创新提供基础和支持。在这次实践活动中，我进行了广泛而深入的研究，并进行了实验验证，以探索图嵌入领域的新理论模型、算法和方法，并将其应用于解决实际问题。首先，我进行了对现有图嵌入方法的系统调研和分析，了解了当前该领域的研究热点和技术趋势。在此基础上，我提出了一种新的理论模型或算法，针对特定的实际问题进行了深入的研究。通过大量的实验和数据分析，我验证了新方法的有效性和性能优势。这些研究和实验的结果为技术应用的创新提供了基础，为进一步的发展奠定了坚实的基础。在本次实践活动中，我还发

现了新的技术趋势和潜力。通过与导师和同行的交流与讨论，我了解到了一些前沿的研究方向和新兴的应用领域。这些发现对于推动技术应用的创新和发展非常重要，它们可以激发新的创意和思路，引领未来的研究和实践方向。此外，本次实践活动还促使我深入思考技术应用的实际问题，并积极探索解决方案。通过将理论模型、算法和方法应用于实际问题的解决，我获得了宝贵的经验和见解。这种将理论与实践相结合的方法，为技术应用提供了更有效的解决方案，并为实际应用的可行性和效果提供了支持。本次实践活动以实际的应用场景为导向，我深入地了解了问题的本质，并提出了创新的解决方案。将理论知识与实际问题相结合，我为技术应用提供了更为有效的解决方案。这种将理论与实践相结合的方法，为技术应用的创新提供了坚实的基础。在本次科研实践中，我产出了丰富的技术成果，这些成果为技术应用的创新和成果转化提供了基础。通过大量的实验、数据分析和验证，我生成了新的技术方法、模型或产品，并在实践中对其进行了迭代和优化。这些技术成果将为进一步的开发和推广奠定基础，促进技术应用的实际落地和商业化。在实践活动中，我也深刻认识到了技术在实际环境中的表现和局限性。通过实际应用场景的反馈，我能够识别出技术的优势和不足之处。这些反馈信息对我进行技术改进和优化起到了重要的指导作用，帮助提高技术应用的效果和适应性。

（二）取得的业绩（代表作）【限填3项，须提交证明原件（包括发表的论文、出版的著作、专利证书、获奖证书、科技项目立项文件或合同、企业证明等）供核实，并提供复印件一份】

1.
公开成果代表作**【论文发表、专利成果、软件著作权、标准规范与行业工法制定、著作编写、科技成果获奖、学位论文等】**

成果名称	成果类别 [含论文、授权专利（含发明专利申请）、软件著作权、标准、工法、著作、获奖、学位论文等]	发表时间/授权或申请时间等	刊物名称/专利授权或申请号等	本人排名/总人数	备注
Representation Enhancement based Cold Start Model for Elastic Compute Service Recommendation	会议论文	2023年07月02日	2023 IEEE International Conference on Web Services (ICWS)	1/7	已发表
Graph Relation Embedding Network for Click-through Rate Prediction	核心期刊	2022年08月01日	Knowledge and Information Systems	1/6	已发表

2.其他代表作【主持或参与的课题研究项目、科技成果应用转化推广、企业技术难题解决方案、自主研发设计的产品或样机、技术报告、设计图纸、软课题研究报告、可行性研究报告、规划设计方案、施工或调试报告、工程实验、技术培训教材、推动行业发展中发挥的作用及取得的经济社会效益等】

（三）在校期间课程、专业实践训练及学位论文相关情况	
课程成绩情况	按课程学分核算的平均成绩： 90 分
专业实践训练时间及考核情况(具有三年及以上工作经历的不作要求)	累计时间： 1 **年**（要求1年及以上） 考核成绩： 91 分（要求80分及以上）
本人承诺	
个人声明：本人上述所填资料均为真实有效，如有虚假，愿承担一切责任，特此声明！ **申报人签名：** [illegible]	

2216017

二、日常表现考核评价及申报材料审核公示结果

日常表现考核评价	非定向生由德育导师考核评价、定向生由所在工作单位考核评价： ☑优秀　　□良好　　□合格　　□不合格 德育导师/定向生所在工作单位分管领导签字（公章）：[illegible]　　年　月　日 中共浙江大学工程师学院委员会
申报材料审核公示	根据评审条件，工程师学院已对申报人员进行材料审核（学位课程成绩、专业实践训练时间及考核、学位论文、代表作等情况），并将符合要求的申报材料在学院网站公示不少于5个工作日，具体公示结果如下： □通过　　□不通过（具体原因：　　　　　　　　　　　） 工程师学院教学管理办公室审核签字（公章）：　　年　月　日

浙 江 大 学 研 究 生 院

攻读硕士学位研究生成绩表

学号：22160173	姓名：吴怡萱	性别：女	学院：工程师学院	专业：计算机技术	学制：2.5年
毕业时最低应获：24.0学分		已获得：25.0学分		入学年月：2021-09	毕业年月：2024-03
学位证书号：1033532024602186			毕业证书号：103351202402600412		授予学位：电子信息硕士

学习时间	课程名称	备注	学分	成绩	课程性质	学习时间	课程名称	备注	学分	成绩	课程性质
2021-2022学年秋季学期	人工智能算法与系统		2.0	96	专业选修课	2021-2022学年春季学期	研究生英语基础技能		1.0	65	公共学位课
2021-2022学年秋季学期	数据科学技术与软件实现		2.0	97	专业学位课	2021-2022学年春夏季学期	研究生英语		2.0	90	公共学位课
2021-2022学年秋季学期	知识图谱导论		2.0	93	专业选修课	2021-2022学年春夏季学期	数据挖掘与机器学习		3.0	93	专业选修课
2021-2022学年秋冬季学期	中国特色社会主义理论与实践研究		2.0	92	公共学位课	2021-2022学年夏季学期	机器学习与数据挖掘工程		2.0	95	专业学位课
2021-2022学年秋冬季学期	数据分析的概率统计基础		3.0	88	专业选修课	2021-2022学年夏季学期	自然辩证法概论		1.0	89	公共学位课
2021-2022学年秋冬季学期	研究生论文写作指导		1.0	90	专业学位课	2021-2022学年春夏季学期	工程伦理		2.0	99	公共学位课
2021-2022学年冬季学期	数据工程实践与案例分析		2.0	99	专业学位课						

说明：1.研究生课程按三种方法计分：百分制，两级制（通过、不通过），五级制（优、良、中、及格、不及格）。

2.备注中“*”表示重修课程。

学院成绩校核章：

（60）

成绩校核人：张梦依

打印日期：2024-04-02

Under consideration for publication in Knowledge and Information
Systems

Graph Relation Embedding Network for Click-through Rate Prediction

Yixuan Wu[1], Youpeng Hu[2], Xin Xiong[3], Xunkai Li[2], Ronghui Guo[2] and Shuiguang Deng[1]
[1]Zhejiang University, Hangzhou, China; [2]Shandong University, Weihai, China; [3]Nanjing University, Nanjing, China

Abstract. Most deep click-through rate (CTR) prediction models utilize a mainstream framework, which consists of the embedding layer and the feature interaction layer. Embeddings rich in semantic information directly benefit the downstream frameworks to mine potential information and achieve better performance. However, the embedding layer is rarely optimized in the CTR field. Although mapped into a low-dimensional embedding space, discrete features are still sparse. To solve this problem, we build graph structures to mine the similar interest of users and the co-occurrence relationship of items from click behavior sequences, and regard them as prior information for embedding optimization. For interpretable graph structures, we further propose Graph Relation Embedding Networks (GREENs), which utilize adapted order-wise graph convolution to alleviate the problems of data sparsity and over-smoothing. Moreover, we also propose a Graph Contrastive Regularization (GCR) module, which further normalizes graph embedding by maintaining certain graph structure information. Extensive experiments have proved that by introducing our embedding optimization methods, significant performance improvement is achieved.

Keywords: Click-through Rate; Graph Embedding; Recommender System; Graph Neural Network

1. Introduction

Whether in online advertising, search engines, or recommender systems (Yi et al. 2019), human computer interaction (Liu et al. 2021), movies (Li et al. 2022), robot service (Liu et al. 2018), and intelligent control (Liu et al. 2019), click-through rate (CTR) prediction tasks are of great research and commercial value, whose result can rank the items returned to a user to maximize the number of clicks.

Received 24 Sep 2021
Revised 12 May 2022
Accepted 18 Jun 2022

Deep learning methods have stronger expressive ability and more flexible structures, which can better handle classification tasks. Instead of traditional methods, a series of representative deep CTR models have been developed by introducing neural networks, such as Wide&Deep (Cheng et al. 2016) and PNN (Qu et al. 2016), *etc.* In recent years, researchers have made various meaningful attempts. DeepFM (Guo et al. 2017) is an end-to-end model with stronger generalization ability and memory ability, which extracts both low- and high-order feature interactions by introducing factorization machine (FM). DIN (Zhou et al. 2017) and DIEN (Zhou et al. 2019), which achieve considerable performance improvements, utilize users' historical click behaviors to mine the distribution and transfer of users' interest respectively as the prior information to provide an estimate of current interest. It can be seen that the inherent prior information of features can effectively improve prediction accuracy.

The above deep CTR models are mainly composed of an input layer, an embedding layer, a feature interaction layer and an output layer, which can be regarded as a joint optimization of representation learning and task-oriented learning. The embedding layer is responsible for densifying the discrete features as their corresponding representations and then passing them to the downstream modules for feature interaction. In practice, densification requires a large number of data to support, while a user's click behaviors or a item's clicked behaviors are too sparse compared to numerous items and users, which brings challenges to learn representations with rich semantic information.

Graph embedding has achieved an excellent effect in the field of embedding representation (Kipf and Welling 2016*b*) (Cui et al. 2020) (Li et al. 2021) (Zhang et al. 2020). By constructing interpretable graph structures, graph learning methods can be applied to the CTR prediction for a more reasonable feature space, as shown in figure 1. Based on click behavior sequences, Graph Intention Network(GIN) (Li et al. 2019) constructs the co-occurrence graph of items and aggregates neighbor nodes by attention mechanism to solve the problems of over-sparse and weak generalization. However, GIN leaves a lot to be desired, such as underutilized relationship information and slow convergence speed.

In addition, the CTR model is prone to overfitting due to a huge amount of parameters for feature extraction and interaction, resulting in its poor generalization ability. To solve it, there is a mainstream solution to introduce regularization terms to enhance generalization ability. Inspired by graph contrastive learning methods such as Deep Graph Infomax (DGI) (Veličković et al. 2018), we can maintain certain graph structure information in a self-supervised way to prevent the overfitting phenomenon. With fewer parameters and no additional labels required, it is considered as an efficient regularization strategy of the graph embedding layer.

Based on previous works and the pain points of CTR prediction, we propose our novel embedding optimization method, namely, Graph Relation Embedding Network (GREEN), on the item co-occurrence graph and the user co-interest graph constructed through click behavior sequences, as shown in figure 3. An adapted order-wise graph convolution is designed in GREEN to aggregate information and provide rich prior information for a more reasonable feature space. Moreover, we propose a Graph Contrastive Regularization (GCR) method to suppress the overfitting phenomenon. It is emphasized that our method is applicable to any deep CTR models for embedding optimization.

In summary, the main contributions of this paper are as follows:

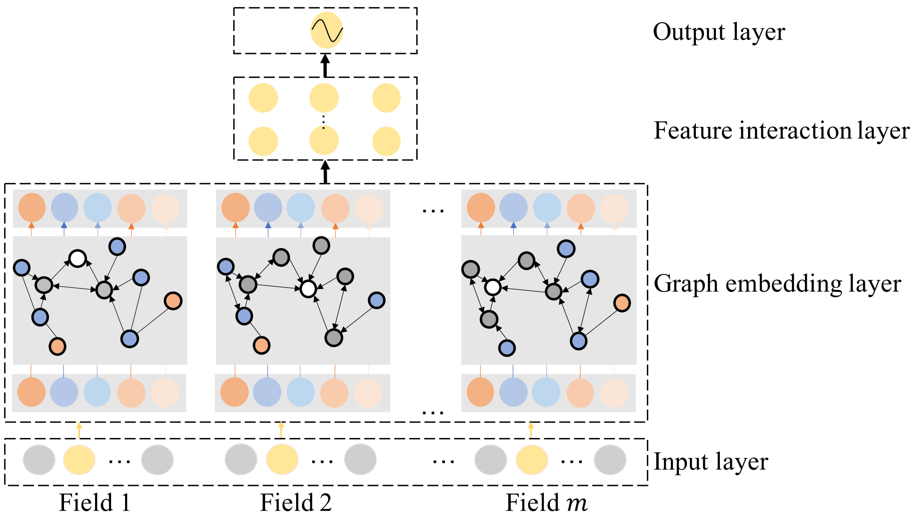


Fig. 1. The deep CTR prediction architecture with the graph embedding layers

- We mine potential relations based on click behaviors and propose a reasonable and interpretable construction strategy of the item co-occurrence graph and the user co-interest graph.
- We propose the Graph Relation Embedding Network. Through the relations among multi-hop neighbors, it can effectively alleviate the data sparsity and learn better representation.
- As a novel regularization method, Graph Contrastive Regularization is proposed to relieve the problem of overfitting.
- Our method is implemented on the public datasets and achieved a considerable performance improvement compared to the baseline.

2. PRELIMINARIES

The datasets for CTR prediction consist of n samples, each of which is represented as $(\mathbf{x}, y)$, where $y \in \{0, 1\}$ represents whether the user clicks the target item in the specific context, $\mathbf{x} = [\mathbf{x}_{field_1}, \mathbf{x}_{field_2}, ..., \mathbf{x}_{field_m}]$, and $\mathbf{x}_{field_i}$ is to describe the feature of user, item, context or others.

Since discrete features are often encoded to sparse one-hot vectors, they are densified through an embedding layer in deep CTR models:

$$(\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_m) = f(\mathbf{x}_{field_1}, \mathbf{x}_{field_2}, ..., \mathbf{x}_{field_m}), \tag{1}$$

where $f(\cdot)$ represents the embedding function which follows the table lookup mechanism, and $\mathbf{e}$ represents the corresponding embeddings. Further, the CTR prediction result is obtained through a feature interaction layer and an output layer:

$$\hat{y} = \sigma(g(\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_m)), \tag{2}$$

where $g(\cdot)$ represents the feature interaction function, such as multilayer perceptrons, $\hat{y}$ is the prediction result of the current data $\mathbf{x}$, and $\sigma(t) = \frac{1}{1+e^{-t}}$. In the training process, the binary cross entropy function is utilized to calculate prediction loss:

$$\mathcal{L}_{BCE} = -\frac{1}{n}\sum_{j}^{n}[y_j \ log \ \hat{y}_j + (1-y_j) \ log \ (1-\hat{y}_j)]. \tag{3}$$

Finally, the end-to-end joint optimization process is carried out by the back propagation.

3. Proposed Approach

3.1. Graph Construction

Compared with the specific framework of Graph Neural Networks (GNNs), a reasonable and interpretable graph structure determines the upper limit of model accuracy to a greater degree. We construct graph structures for users and items respectively to enrich their embeddings for both attribute information and the corresponding topology. Embeddings richer in semantic information directly benefit the downstream frameworks for feature interaction to achieve better performance. Graph construction is based on the following accepted assumptions:

- Users who click on the same item have a degree of interest similarity. The degree of interest similarity of users is related to the number of the same items they clicked in general.
- Items which are continuously clicked by the same user have a certain co-occurrence relation. The degree of co-occurrence relation of items depends on the times of being clicked by the same users continuously.

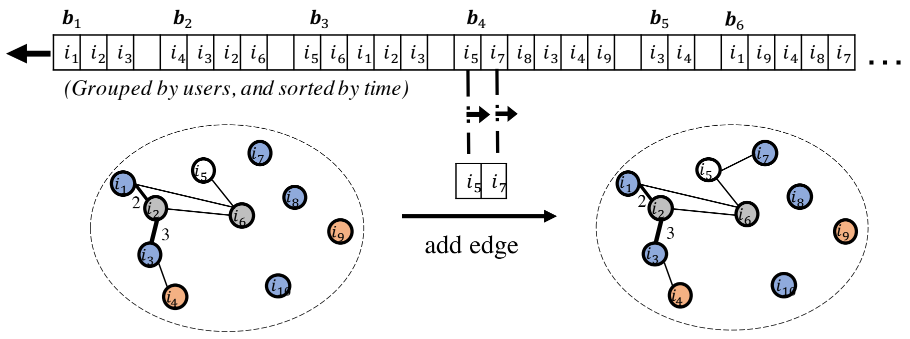
Given the item set $\mathbf{I} = \{i_1, i_2, ..., i_N\}$, the user set $\mathbf{U} = \{u_1, u_2, ..., u_M\}$, and the click behaviors $\mathbf{b}_j = [i_{j_1}, i_{j_2}, ..., i_{j_k}]$ of each user u_j, where N and M represent the total number of items and users respectively, and the length of the click behaviors k is different for different users, we construct a user co-interest graph $\mathbf{G}_u = (\mathbf{U}, \mathcal{E}_u)$ and an item co-occurrence graph $\mathbf{G}_i = (\mathbf{I}, \mathcal{E}_i)$, where $\mathcal{E}_u$ and $\mathcal{E}_i$ are the weighted edge sets of the user co-interest graph and the item co-occurrence graph, respectively.

The construction of the item co-occurrence graph is shown in figure 2(a). Referring to (Li et al. 2019), we iterate through each user's click behavior sequences in chronological order to connect items that have been continuously clicked. If the two items are connected for the first time, their weight is set to one, otherwise, their weight is increased by one. A bigger weight between any two items illustrates it is more possible for them to be continuously clicked again.

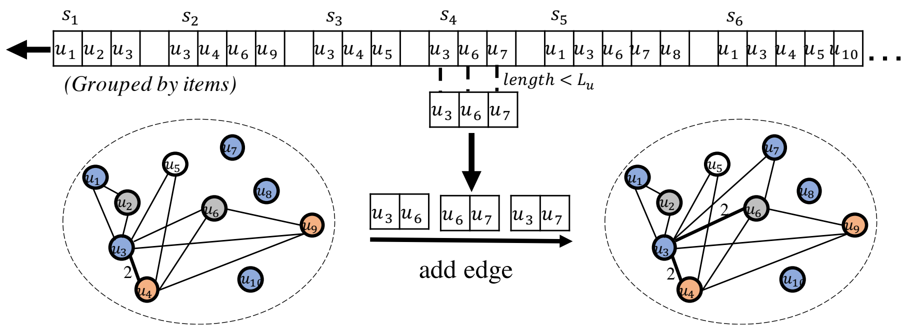
On the other side, the user co-interest graph is shown in figure 2(b), where s_j represents the user set clicked the same item i_j. We connect all users who have clicked the same non-popular item, whose clicked number is smaller than maximum length L_u, for popular items lead to considerable meaningless relationships of users constructed and increase the complexity of the graph. If they have been connected, the weight is increased by one. In this way, a bigger weight between users means more similar click interests.

Based on this, we obtain the weighted adjacency matrices $\mathbf{A}_i \in \mathbb{R}^{N\times N}$ and $\mathbf{A}_u \in \mathbb{R}^{M\times M}$ to describe the connection relationship and strength among items

[t]


b_1 b_2 b_3 b_4 b_5 b_6
(Grouped by users, and sorted by time)
add edge

(a) Item co-occurrence graph



(b) User co-interest graph

Fig. 2. An illustration of graph construction, where b_i represents the sorted click behavior sequences of user i, s_j represents the user set clicked item j, and j will be regarded as a popular item if the length of s_j is longer than L_u

or users. It is emphasized that not only can the user embedding and the item embedding be optimized through our model, but also others with latent graph structures are well applied.

3.2. Graph Relation Embedding Network

In the item co-occurrence graph and the user co-interest graph, connected nodes have similar clicked intentions or click interests. For example, when the user u_j clicks on the item i_m, it is extremely possible that u_j will click on another item that has a co-occurrence relationship with i_m, and i_m will be clicked by another user that has a co-interest relationship with u_j. It will be reflected in the relative position in the feature space, where the embedding representations of co-occurrence items or co-interest users are more closer through the neighbor aggregation of nodes on the graph.

Graph Relation Embedding Network architecture is shown in figure 3. Taking item feature as an example, we define the initial embedding matrix $\mathbf{X} \in \mathbb{R}^{N\times d}$, the weighted adjacency matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ of the graph structure, and the

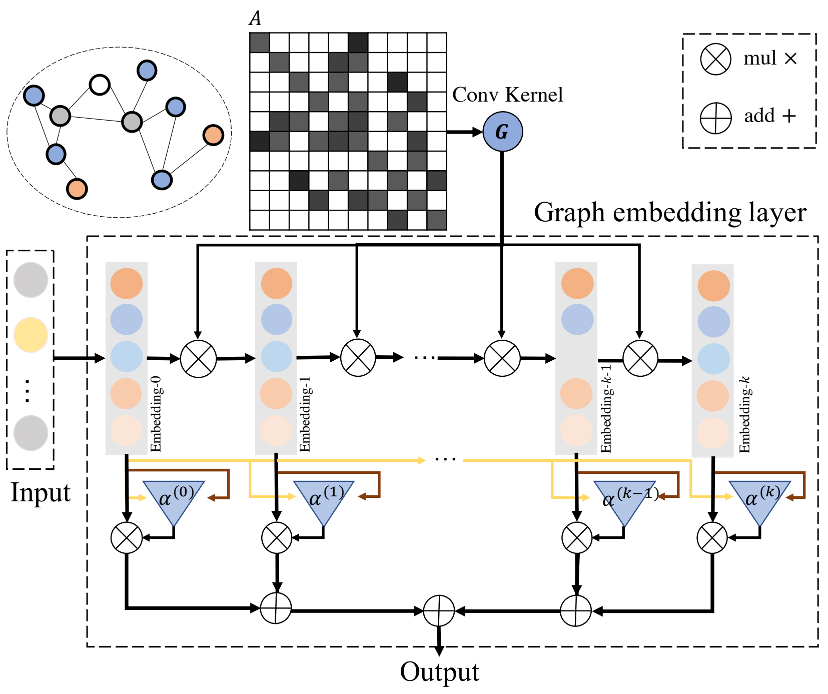


Fig. 3. Specific implementation of the Graph Relation Embedding Network

edges' degree matrix $\mathbf{D} \in \mathbb{R}^{N\times N}$, where d is the dimension of embeddings and $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{i,j}$. The Graph Convolutional Network (GCN) was proposed in (Kipf and Welling 2016*a*):

$$\mathbf{H} = \mathbf{G}\mathbf{X}\boldsymbol{\Theta} = \widetilde{\mathbf{D}}^{-1/2}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-1/2}\mathbf{X}\boldsymbol{\Theta}, \tag{4}$$

where $\widetilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$, $\widetilde{\mathbf{D}} = \mathbf{D} + \mathbf{I}_N$, $\mathbf{I}_N$ is an identity matrix whose dimension is the number of nodes N, and $\boldsymbol{\Theta}$ is a learnable right multiplication matrix, which is used to map the feature space of $\mathbf{X}$ to a new feature space. In Eq. (4), the graph convolution kernel $\mathbf{G} = \widetilde{\mathbf{D}}^{-1/2}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-1/2}$ is used for feature aggregation of adjacent nodes.

For the CTR task, the embedding representations of users and items have exclusive feature space. Therefore, it is unnecessary to perform redundant and repeated feature space mapping, or introduce considerable optional learnable parameters resulting in the inference slowdown, which is also verified in (Wu et al. 2019). From the formula analysis, in our work k order graph convolution is defined as:

$$\mathbf{X}^{(k)} = \mathbf{G}\mathbf{X}^{(k-1)}, \tag{5}$$

where $\mathbf{X}^{(0)} = \mathbf{X}$ which is the initial embedding matrix, and $\mathbf{X}^{(k)}$ is the feature aggregation of $\mathbf{X}$ through k times.

When we use multi-order graph convolution to perform the aggregation representation of embedding, we will inevitably encounter over-smoothing problems,

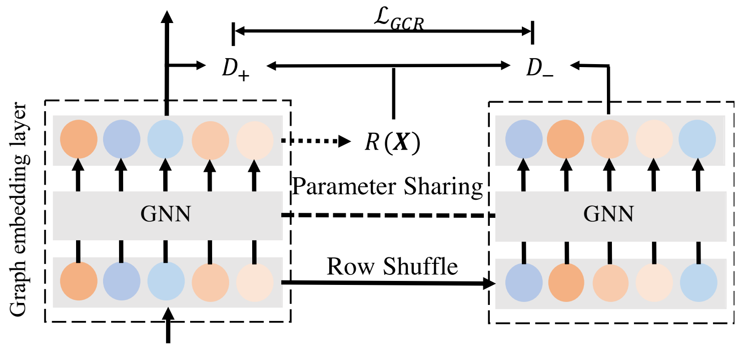


Fig. 4. An illustration of the Graph Contrastive Regularization

i.e. the phenomenon that all node embeddings tend to be consistent, making the CTR prediction inaccurate. In order to solve the problem, we introduce adapted order-wise weights. Inspired by the weight of attention defined in (Li et al. 2019), the weight calculation corresponding to the k order graph convolution is:

$$\alpha^{(k)} = \frac{1}{N}\sum_{j=1}^{N}\sigma[(\mathbf{X}^{(k)}\|\mathbf{X}\|(\mathbf{X}^{(k)} - \mathbf{X})\|(\mathbf{X}^{(k)} \odot \mathbf{X}))\mathbf{W}], \tag{6}$$

where $\cdot\|\cdot$ represents matrix concatenation, $\sigma(\cdot)$ is the sigmoid function, $\odot$ denotes the element-wise product, and $\mathbf{W} \in \mathbb{R}^{4d\times 1}$ is the trainable parameter. Then, the final output of the GREEN is:

$$\mathbf{X}_{out} = \sum_{j=0}^{k}\alpha^{(j)}\mathbf{G}\mathbf{X}^{(j)}. \tag{7}$$

Eq. (7) brings additional time complexity $O(kdM)$ to base model by sparse computing, where M represents the number of edges, and $M >> k, d$. Reasonable trimming for edges can effectively accelerate the inference.

Applying the architecture of GREEN, each embedding is learned by more sufficient data through the graph structure, which greatly alleviates the problem of data sparsity. Therefore, by truncating outdated historical behaviors, the graph is adapted to the real-time relationship changed rapidly, and the model can even obtain higher accuracy with less data.

3.3. Graph Contrastive Regularization

Inspired by the Deep Graph Informax (DGI) (Veličković et al. 2018) model based on the contrastive paradigm, a regularization method based on graph contrastive learning is proposed to further suppress overfitting, namely, Graph Contrastive Regularization (GCR), as shown in figure 4.

The core idea of contrastive learning is to find three components from the original data: positive sample, negative sample, and anchor. We randomly shuffle the initial embedding matrix $\mathbf{X}$ of nodes to generate fake features $\widetilde{\mathbf{X}}$, and set $\mathbf{x}_i$ and $\widetilde{\mathbf{x}}_i$ as the real and fake feature of i-th node. We input $\widetilde{\mathbf{X}}$ into the same

GREEN model to obtain the output representation of the embedding layer:

$$\widetilde{\mathbf{X}}_{out} = \sum_{j=0}^{k} \alpha^{(j)} \mathbf{G} \widetilde{\mathbf{X}}^{(j)}, \tag{8}$$

where $\widetilde{\mathbf{X}}_{out}$ represents the fake feature matrix. In addition, we use the mean function as the readout step to extract the graph embedding representation as the anchor:

$$R(\mathbf{X}) = \sigma(\frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_i), \tag{9}$$

where $\sigma(\cdot)$ is the sigmoid function. Furthermore, a bilinear scoring function is utilized as the discriminator:

$$D(\mathbf{x}_i, \mathbf{r}) = \sigma({\mathbf{x}_i}^T \mathbf{W} \mathbf{r}), \tag{10}$$

where $\mathbf{r}$ represents the anchor introduced in Eq.(9), $\sigma(\cdot)$ is the sigmoid function, and $\mathbf{W} \in \mathbb{R}^{d \times d}$ is a learnable scoring matrix to generate the sample's score for $(\mathbf{x}_i, \mathbf{r})$. Finally, we use noise contrastive estimation(NCE) loss (Mnih and Kavukcuoglu 2013):

$$\begin{aligned} \mathcal{L}_{GCR} = -\frac{1}{2M}(&\sum_{v_i \in S} \mathbb{E}_{(\mathbf{X},\mathbf{A})}[log\ D(\mathbf{x}_{out\ i}, R(\mathbf{X}_{out}))] \\ &+ \sum_{v_j \in S} \mathbb{E}_{(\widetilde{\mathbf{X}},\mathbf{A})}[1 - log\ D(\widetilde{\mathbf{x}}_{out\ j}, R(\mathbf{X}_{out}))]), \end{aligned} \tag{11}$$

where $S \subset \mathcal{V}$ is a randomly selected subset of the node set $\mathcal{V}$. To balance the consumption of additional resources and the degree of regularization, the size $|S|$ is adjusted depending on datasets characteristics. By Eq.(11), we effectively maximizes the mutual information between $x_{out\ i}$ and $R(X_{out})$, *i.e.* the JS divergence between the joint distribution and the product of marginal distribution.

$\mathcal{L}_{GCR}$ can be directly added to the base model loss Eq. (3) for integrated end-to-end learning:

$$\mathcal{L} = \mathcal{L}_{BCE} + w\mathcal{L}_{GCR}, \tag{12}$$

where w is the weight of the GCR module as regularization term, which balances the graph contrastive learning and CTR prediction. By sharing parameters of GREEN, the two tasks complement information and promote learning together, improving the generalization ability of the model. Specifically, in the training process, all trainable parameters in the model are optimized by minimizing $\mathcal{L}$. In the test process, the prediction result is obtained through the main task without GCR.

Overall, the presence of ancillary loss $\mathcal{L}_{GCR}$ has several advantages. First, the introduction of GCR will maintain certain graph structure information to a certain extent. Secondly, the multi-task learning paradigm can relieve the overfitting phenomenon because it can improve the model's robustness to unseen data (Pironkov et al. 2016). Finally, as an ancillary task, it does not impose any computational burden on the model application.

Table 1. Statistics of the datasets

Dataset	Users	Items	Categories	Samples
Amazon (Electro)	192,403	63,001	801	1,689,188
Amazon (Movies and TV)	123,960	50,052	29	1,697,533
MovieLens	138,493	27,278	21	20,000,263

4. Experiments

4.1. Experimental Settings

4.1.1. Datasets

The statistical information of the datasets is shown in table 1, and the description is as follows:

Amazon[1] (He and McAuley 2016): is used as the benchmark dataset with pretty rich click behaviors for CTR prediction, which contains product reviews and metadata. We use two subsets: **Electronics** and **Movies and TV**, to vertify the effect of our embedding optimization method. We group the samples by users, and each user's click behaviors can be described as $(b_1, b_2, ..., b_n)$. Our goal is to predict each user's n-th click behavior based on the past $n-1$ behaviors.

MovieLens[2] (Harper and Konstan 2015): is a dataset used to describe users' ratings ranging from 0 to 5, which is treated as a binary classification problem here, where the click data with the rating no less than 4 are regarded as positive samples, and others are regarded as negative. We group the samples by each user to predict his n-th click behavior following the Amazon dataset. In order to prevent the over-rich historical information, we take the latest 10 historical click behaviors as users' latent interests in our experiments to enhance the prediction difficulty.

4.1.2. Baselines

We introduce GREEN and GCR successively on five basic models to verify our methods.

Wide & Deep (Cheng et al. 2016): combined by a wide component and a deep component is proposed to capture both low-order and high-order feature interactions, which takes both memory ability and generalization ability of the model into account.

PNN (Qu et al. 2016): introduces a product layer after the embedding layer to better extract high-order feature interactions.

Deep Crossing (Shan et al. 2016): uses multiple residual units to mine the relationship between features, instead of explicitly interacting features.

DeepFM (Guo et al. 2017): combines the advantages of factorization machines (FMs) and deep learning networks (DNNs) to shorten the convergence time while ensuring accuracy, where FM extracts low-order feature interactions

[1] http://jmcauley.ucsd.edu/data/amazon/
[2] https://grouplens.org/datasets/movielens/20m/

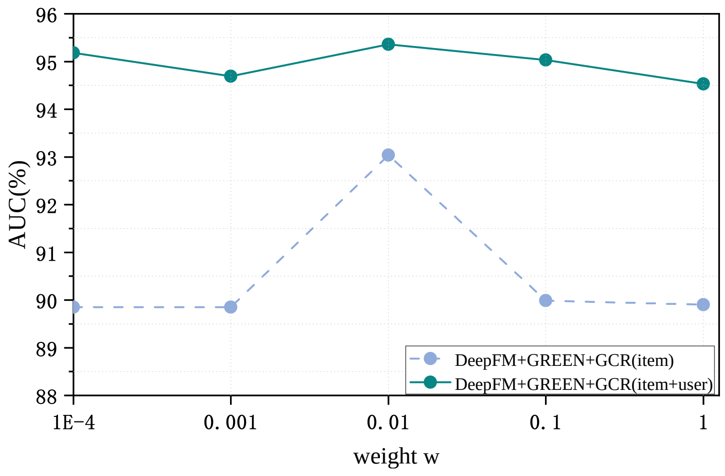


Fig. 5. The relation curve between the weight w of the GCR module mentioned in Eq.(12) and AUC on Amazon (Electro)

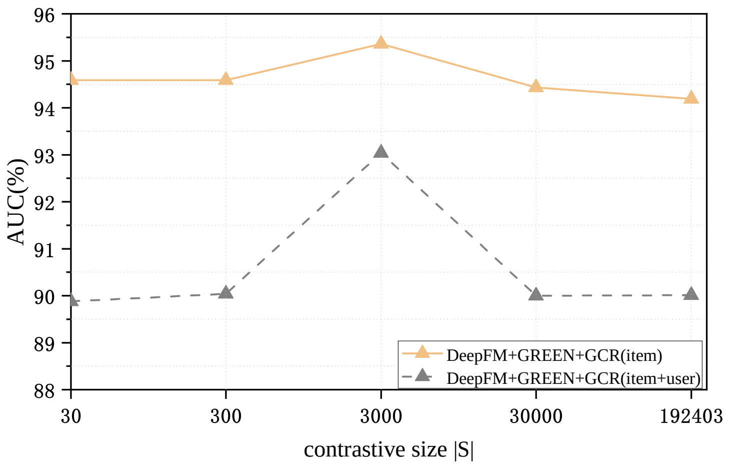


Fig. 6. The relation curve between the contrastive size $|S|$ of the GCR modul mentioned in Eq.(11) and AUC on Amazon (Electro), where the total number of nodes is 192403

through multiple inner product units and a linear unit, and DNN extracts high-order interactions among features through MLP layers.

DIN (Li et al. 2019): introduces the attention mechanism to extract the users' latent interests from their own historical behaviors. We combine the feature of historical behaviors with user profile feature, item feature, context feature, *etc*, and then input them to MLP for end-to-end learning.

4.1.3. Matrics

In the field of CTR prediction, AUC is used to evaluate the effectiveness of models (Fawcett 2006). Since the goal of CTR prediction is to sort the candidate

Table 2. Prediction results on the Amazon (Electro) dataset

	Item		Item&User	
	AUC	RelaImpr	AUC	RelaImpr
Wide & Deep	86.09	0.00%	86.09	0.00%
+GREEN	89.30	8.89%	93.19	19.67%
+GCR	89.30	8.89%	93.24	19.81%
PNN	86.54	0.00%	85.83	0.00%
+GREEN	90.18	9.96%	95.85	27.97%
+GCR	90.30	10.29%	95.88	28.05%
Deep Crossing	86.72	0.00%	85.85	0.00%
+GREEN	89.33	7.11%	93.43	21.14%
+GCR	89.43	7.38%	93.47	21.26%
DeepFM	86.93	0.00%	86.38	0.00%
+GREEN	89.87	7.96%	94.94	23.52%
+GCR	89.95	8.17%	95.36	24.68%
DIN	87.15	0.00%	87.27	0.00%
+GREEN	91.37	11.35%	94.16	18.48%
+GCR	91.45	11.57%	94.34	18.96%

items of each user, there are differences among different users, such as some users have a higher click rate or often give higher scores to items. Therefore, we use GAUC proposed by (Zhu et al. 2017) as our AUC matric:

$$AUC = \frac{\sum_{i=1}^{n} \#impression_i \times AUC_i}{\sum_{i=1}^{n} \#impression_i}. \tag{13}$$

Furthermore, RelaImpr (Yan et al. 2014) is used to measure relative improvement:

$$RelaImpr = (\frac{AUC(objective\ model) - 0.5}{AUC(base\ model) - 0.5} - 1) \times 100\%. \tag{14}$$

4.1.4. Implementation

To verify the validity, the hyperparameters for all models are consistent, followed by (Zhou et al. 2017). The models are learned by the Adam optimizer, the learning rate is set to 0.001, and its decay rate is 0.9 per 336000 samples. For all datasets, the training batch size is set to 32, the testing batch size is set to 512, the embedding dimension d is set to 128, the maximum length L_u is set to 40, the order k of GREEN is set to 4, and the sigmoid function is used as the activation function.

For the GCR module, the contrastive size $|S|$ is relevant to the scale of the dataset, and the weight w is relevant to the model. For Amazon (Electro), $|S|$ and w are set to 3000 and 0.01, respectively. For MovieLens, $|S|$ is set to 1000, and w is set to 2 for all except 0.001 for Deep Crossing. And for Amazon (Movies and TV), $|S|$ is set to 25000, and w is set to 0.01, 0.001, 0.1, 0.001, 0.1 for Wide & Deep, PNN, Deep Crossing, DeepFM and DIN, respectively. In order

Table 3. Prediction results on the MovieLens dataset

	Item		Item&User	
	AUC	RelaImpr	AUC	RelaImpr
Wide & Deep	71.73	0.00%	71.00	0.00%
+GREEN	72.83	5.06%	71.59	2.81%
+GCR	72.83	5.06%	72.21	5.76%
PNN	73.50	0.00%	73.92	0.00%
+GREEN	74.45	4.04%	74.54	2.59%
+GCR	74.55	4.47%	75.40	6.19%
Deep Crossing	72.41	0.00%	72.00	0.00%
+GREEN	73.59	5.27%	72.77	3.50%
+GCR	73.68	5.67%	72.79	3.59%
DeepFM	73.66	0.00%	75.74	0.00%
+GREEN	74.73	4.52%	76.06	1.24%
+GCR	75.22	6.59%	76.07	1.28%
DIN	74.53	0.00%	73.97	0.00%
+GREEN	75.76	5.01%	75.85	7.84%
+GCR	75.83	5.30%	76.04	8.64%

to study the performance impact of the two parameters in the GCR module, we conduct experiments by introducing GREEN and GCR in deepfm on the Amazon (Electro). As can be seen from figure 5, by fixing $|S|$ to 3000, AUC varies with w, and the model performs best when w is set to 0.01. Similarly, by fixing w to 0.01, the model performs best when $|S|$ is set to 3000 as shown in figure 6.

4.2. Ablation Study

Based on Wide & Deep, PNN, Deep Crossing, DeepFM and DIN, we extract item features and user features of click samples in turn to verify the validity of the item co-occurrence graph and the user co-interest graph applied GREEN architecture and GCR successively, and the experimental results are shown in table 2, table 3, and table 4. On all base models and datasets, GREEN can provide consistent and significant performance improvement, even *relaImpr* is up to 27.97%. Compared to MovieLens with richer samples, GREEN has more impressive improvements on Amazon with sparse samples, which proves that our model can relieve the problem of data sparsity. Moreover, the introduction of GCR can achieve further performance improvements.

4.3. Comparative Study

To further verify the effectiveness of our methods, we compare with the existing CTR models for feature optimization. DUSIN (Kim et al. 2021) and DDIL(Zhang et al. 2021) are both sequential recommendation models for CTR prediction,

Table 4. Prediction results on the Amazon (Movies and TV) dataset

	Item		Item&User	
	AUC	RelaImpr	AUC	RelaImpr
Wide & Deep	88.44	0.00%	87.43	0.00%
+GREEN	92.58	10.77%	94.51	18.92%
+GCR	92.60	10.82%	94.69	19.40%
PNN	89.33	0.00%	89.40	0.00%
+GREEN	93.19	9.81%	96.14	17.11%
+GCR	93.25	9.97%	96.14	17.11%
Deep Crossing	89.50	0.00%	88.46	0.00%
+GREEN	92.54	7.70%	94.73	16.30%
+GCR	92.55	7.72%	94.83	16.56%
DeepFM	90.13	0.00%	89.27	0.00%
+GREEN	93.04	7.25%	95.24	15.20%
+GCR	93.06	7.30%	95.32	15.40%
DIN	89.47	0.00%	88.60	0.00%
+GREEN	93.37	9.88%	94.84	16.17%
+GCR	93.45	10.08%	95.03	16.66%

which model behavior sequences to optimize the feature of user interest. DUSIN extracts and segments users' dynamic interests by considering the user's own historical sequence and potential interests of similar users. DDIL divides user interests into local sessions and global sessions, which are used to capture users' short-term dynamic interests and long-term interests, respectively. In addition, DDIL learns the heterogeneous behaviors within the sessions with consistency learning. The two models both concatenate user feature, item feature, and the optimized feature of user interest into a multiple layer perception (MLP).

The experimental results in table 5 show that, in most cases, the model performs best by introducing GREEN rather than DUSIN or DDIL. Moreover, it can bring further performance improvement when utilizing our proposed embedding optimization method on the item feature and user feature of DUSIN and DDIL.

4.4. Historical Behavior Truncation Study

We restrict the number of click behaviors of each user as l for sparse data on Amazon (Electro), *i.e.* retain the latest l historical behaviors $(b_{n-l}, b_{n-l+1}, ..., b_{n-1})$ in the behavior sequences $(b_1, b_2, ..., b_{n-1})$. The truncation parameters and experimental results are shown in table 6, where m represents the maximum number of behaviors, and $r\%$ represents the proportion of samples retained in the dataset. It is obvious that GREEN achieves a greater performance improvement for more sparse click behaviors. Through the experimental results, we observe that for lower m, the accuracy of the base model decreases due to data sparsity, but the GREEN-based model shows a remarkable upward trend. It shows that the GREEN framework is less sensitive to data sparsity, because relatively com-

Table 5. Prediction results of comparative study on three CTR datasets

	Item		Item&User	
Amazon (Electro)	AUC	RelaImpr	AUC	RelaImpr
MLP	85.78	0.00%	84.92	0.00%
+DUSIN	86.98	3.35%	87.14	6.36%
+DDIL	87.12	3.75%	85.91	2.84%
+GREEN	89.28	9.78%	93.02	23.20%
+DUSIN +GREEN	91.53	16.07%	94.32	26.92%
+DDIL +GREEN	88.33	7.13%	93.08	23.37%
MovieLens	AUC	RelaImpr	AUC	RelaImpr
MLP	71.65	0.00%	70.69	0.00%
+DUSIN	71.01	-2.96%	70.30	-1.88%
+DDIL	75.08	15.84%	74.93	20.49%
+GREEN	72.90	5.77%	71.67	4.74%
+DUSIN +GREEN	72.95	6.00%	72.12	6.91%
+DDIL +GREEN	75.95	19.86%	76.29	27.07%
Amazon (Movies and TV)	AUC	RelaImpr	AUC	RelaImpr
MLP	88.53	0.00%	87.15	0.00%
+DUSIN	89.22	1.79%	87.80	1.75%
+DDIL	89.82	3.35%	86.13	-2.75%
+GREEN	92.49	10.28%	94.56	19.95%
+DUSIN +GREEN	93.88	13.89%	95.35	22.07%
+DDIL +GREEN	94.79	16.25%	95.07	21.32%

plete graph structures can be constructed with limited historical behaviors, and even the graphs constructed by more real-time data can achieve more excellent performance.

4.5. Graph Convolution Order Study

We conduct experiments on Amazon (Electro) about different convolution orders k on the graph convolution architecture and the GREEN architecture respectively, and the experimental results are shown in figure 7. Graph convolution method utilizes the final order as node representations after multiple aggregation, where GREEN introduces adaptive order-wise weights among different orders. On the item co-occurrence graph, when the order k is more than 3, the accuracy of the graph convolution method decreases due to over-smoothing, while the accuracy of GREEN has been continuously improved with the increasing order. On the two graphs, the accuracy of the graph convolution method has been maintained at a low level with the increase of k, where GREEN achieves considerable performance by learning excellent adaptive order-wise weights among multiple orders.

Table 6. Historical behavior truncation experimental results on Amazon (Electro)

$m(r\%)$	20(90.10%)	12(80.49%)	9(71.90%)	7(61.92%)	6(54.47%)	5(44.25%)	4(29.50%)	3(14.75%)
Wide & Deep	**85.73**	85.56	85.29	85.14	85.06	84.73	83.78	82.05
+GREEN	93.13(+7.4)	93.40(+7.84)	93.34(+8.05)	93.11(+7.97)	93.16(+8.10)	**93.51**(+8.78)	92.46(+8.68)	91.66(+**9.61**)
PNN	**86.37**	86.24	86.06	85.81	85.69	85.31	84.50	83.24
+GREEN	95.86(+9.49)	96.19(+9.95)	96.37(+10.31)	96.51(+10.70)	96.85(+11.16)	97.16(+11.85)	97.72(+13.22)	**98.13**(+**14.89**)
Deep Crossing	**86.23**	86.15	85.96	85.82	85.41	85.26	84.43	82.94
+GREEN	93.00(+6.77)	92.99(+6.84)	92.87(+6.91)	93.15(+7.33)	93.37(+7.96)	94.07(+8.81)	93.86(+9.43)	**94.52**(+**11.58**)
DeepFM	**86.65**	86.57	86.60	86.13	86.04	85.56	84.83	83.69
+GREEN	95.22(+8.57)	95.69(+9.12)	95.89(+9.29)	96.24(+10.11)	96.49(+10.45)	96.96(+11.40)	97.19(+12.36)	**97.63**(+**13.94**)
DIN	**87.00**	86.65	86.67	86.25	85.98	85.52	84.58	83.15
+GREEN	94.08(+7.08)	94.29(+7.64)	94.24(+7.53)	94.44(+7.53)	94.66(+8.46)	94.61(+9.09)	**94.98**(+10.40)	94.36(+**11.21**)

4.6. Overfitting Analysis

Figure 8 illustrates the trend of training loss and test loss on Amazon (Electro). Compared with base models, GREEN leads to a rapid drop in loss value, which is significantly lower than the original. It can be seen from figure 8, when the number of training steps reaches 160000 to 180000, the training loss of almost all models decreased, while their test loss increased, which shows that there is an overfitting phenomenon. By introducing GREEN and GCR, the phenomenon is alleviated effectively, and the degree of separation between train and test loss curves is reduced, which proves that it can alleviate the phenomenon of overfitting. Moreover, GCR further reduces the minimum loss to obtain better accuracy base on GREEN.

4.7. Application Analysis

The inference time for the test set and the trainable parameter quantity of our models are shown in table 7. The inference time is measured in a single NVIDIA GTX 2080Ti GPU. Neither GREEN nor GCR will significantly increase the number of learnable parameters. GREEN sacrifices a certain amount of time to bring significant performance improvement, which promotes the accuracy of

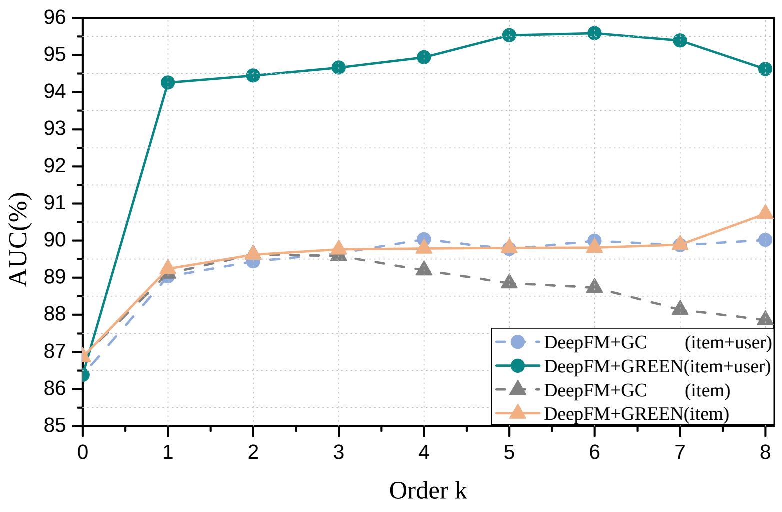


Fig. 7. The relation curve between order k and AUC on Amazon (Electro), where GC represents the graph convolution method without adapted order-wise weights

CTR prediction to a new level. Moreover, due to the independence of GCR, it do not affect the inference time during the test process.

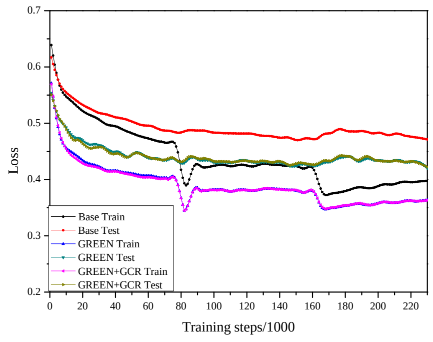
5. Related Work

5.1. Deep CTR

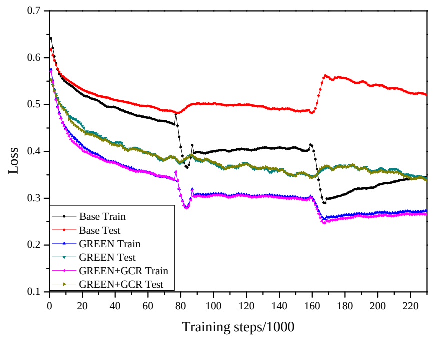
Many methods for feature interaction appeared in the field of CTR prediction, such as Logistic Regression (LR) (Richardson et al. 2007), Factorization Machine (FM) (Rendle and Schmidt-Thieme 2010), or Field-aware Factorization Machines (FFM) (Juan et al. 2016). Benefiting from the advantages of deep learning, some combined models based on deep neural networks have greatly improved the accuracy of CTR prediction. Product-based Neural Network (PNN) (Qu et al. 2016) utilizes product layers for feature intersection. Wide & Deep architecture (Cheng et al. 2016) takes both memory ability and generalization ability of the model into account. DeepFM (Guo et al. 2017) uses a factorization machine to enhance the capability of feature interaction. Deep Interest Network (DIN) (Zhou et al. 2017) introduces the attention mechanism to mine users' interest, and Deep Interest Evolution Network (DIEN) (Zhou et al. 2019) further excavates the transfer of users' interest to assist prediction. Our optimization method for the embedding layer is universal and compatible with all the above models.

5.2. Graph Neural Network

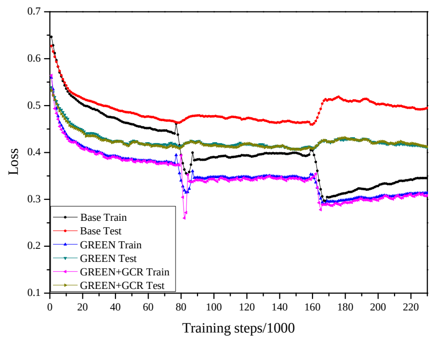
Graph Neural Networks (GNNs) received unprecedented attention in recent years because of its efficient performance (Xu et al. 2018), such as graph convolutional networks (GCNs) (Kipf and Welling 2016*a*), graph attention networks (GATs)



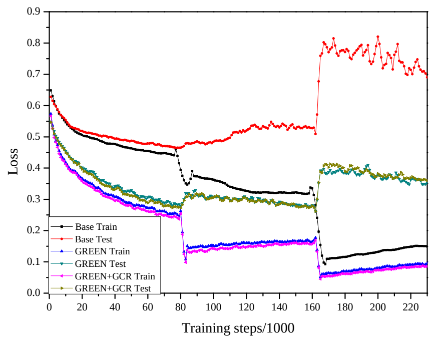
(a) Wide & Deep(item)



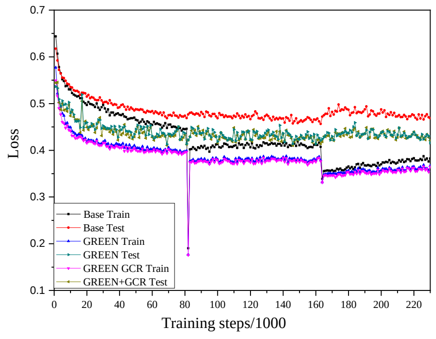
(b) Wide & Deep(item+user)



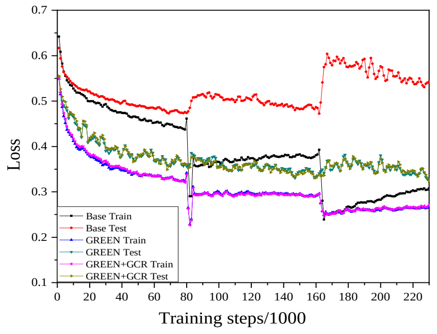
(c) PNN(item)



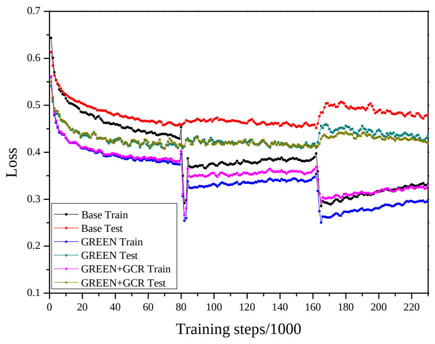
(d) PNN(item+user)



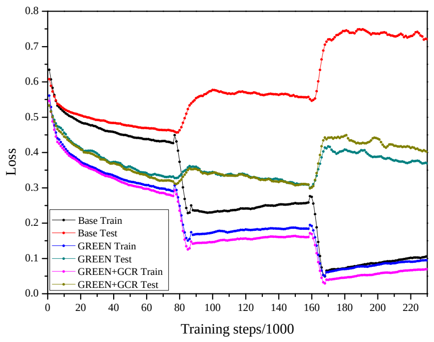
(e) Deep Crossing(item)



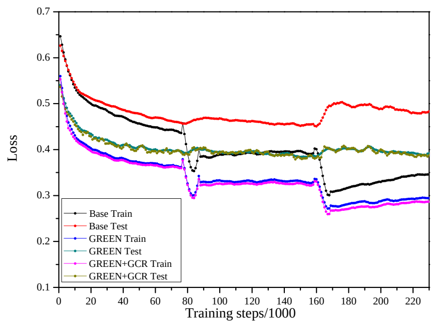
(f) Deep Crossing(item+user)



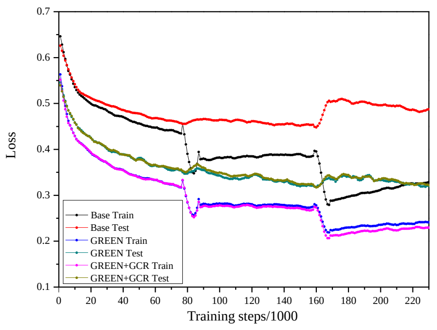
(g) DeepFM(item)



(h) DeepFM(item+user)



(i) DIN(item)



(j) DIN(item+user)

Fig. 8. The curve of loss on Amazon (Electro)

(Veličković et al. 2017), and GraphSAGE (Hamilton et al. 2017). They are based on the methods of neighbor aggregation to integrate the node information to optimize downstream tasks (Zhou et al. 2018). Graph-based tasks have been expanded to include representation learning(Kipf and Welling 2016*b*) (Cui et al. 2020), clustering (Bo et al. 2020) (Li et al. 2020), and link prediction (Chen et al. 2020), *etc.* We design the GNN framework to optimize the learning ability of the embedding layer and introduce various skills to minimize the inference time and solve the over-smoothing problem.

On the other side, graph contrastive learning prospers in the field of graph embedding. Deep Graph Infomax (DGI) (Veličković et al. 2018) introduces the work of Deep Infomax (DIM) (Hjelm et al. 2018) into the graph field. DGI constructs negative samples through feature shuffle and learns better node embeddings by maximizing mutual information between local representations and global graph representations. Inspired by this, we propose a graph contrastive regularization method for the deep CTR model to maintain a certain graph structure and suppress the overfitting problem.

5.3. Graph on Recommendation

Recommender systems (Yi et al. 2019) (Liu et al. 2020) use certain algorithms to solve the problem of information overload, and filter out different candidate sets for different users quickly and individually, which are mostly used in search engines, movies (Li et al. 2022) (Zhang et al. 2015), e-commerce (Zhao et al. 2015) and other fields. Graph learning has a wide range of meaningful applications (Zhang et al. 2019) (Fan et al. 2019), and researchers have tried to introduce them into the recommendation field. Taking collaborative filtering as an example, its core information, the sparse user-item matrix, is a ready-made graph structure. Therefore, the inherent information can be mined in the form of graphs, such as NGCF (Wang et al. 2019) and the faster and lighter Light-GCN (He et al. 2020).

In the field of CTR, Graph Intention Network (GIN) (Li et al. 2019) utilizes historical click behaviors to construct the co-occurrence graph of items, and uses the GAT to aggregate neighbor nodes to solve the problems of sparseness and weak generalization. However, GIN leaves a lot to be desired, such as under-

Table 7. The inference time for the test set and the trainable parameter quantity of our models for Amazon (Electro)

Model	Times(s)	Trainable Params
Wide & Deep	5.960	28.815M
+GREEN(k=1)	23.980	28.827M
+GREEN(k=2)	37.305	28.827M
+GREEN(k=3)	52.966	28.828M
+GREEN(k=4)	60.050	28.829M
+GCR	61.246	28.849M
PNN	5.809	28.826M
+GREEN(k=1)	21.747	28.826M
+GREEN(k=2)	32.490	28.827M
+GREEN(k=3)	48.207	28.828M
+GREEN(k=4)	60.318	28.829M
+GCR	61.170	28.849M
Deep Crossing	7.913	28.923M
+GREEN(k=1)	20.659	29.088M
+GREEN(k=2)	34.208	29.089M
+GREEN(k=3)	47.789	29.090M
+GREEN(k=4)	60.914	29.091M
+GCR	61.845	29.111M
DeepFM	7.892	28.815M
+GREEN(k=1)	23.671	28.826M
+GREEN(k=2)	33.735	28.827M
+GREEN(k=3)	47.778	28.828M
+GREEN(k=4)	59.766	28.828M
+GCR	59.055	28.849M
DIN	9.755	28.869M
+GREEN(k=1)	22.956	28.881M
+GREEN(k=2)	37.114	28.882M
+GREEN(k=3)	47.106	28.882M
+GREEN(k=4)	59.447	28.883M
+GCR	60.337	28.904M

utilized relationship information, considerable parameters, and slow convergence speed. Our work is mainly to carry on a series of research and optimization to solve the weakness of the graph method in the field of CTR prediction.

6. Conclusion

In this paper, we offer the guidance of graph construction with interpretability, introducing graph learning methods into the field of CTR prediction. To take advantage of the prior relationship mined in the graphs, we propose a novel embedding framework named Graph Relation Embedding Network (GREEN), which utilizes multi-order graph convolution and adaptive order-wise weighting to aggregate information for a more reasonable feature space. Moreover, a Graph Contrastive Regularization (GCR) module is designed to further normalize graph embedding by maintaining certain graph information. We conduct extensive experiments and the results verify that our methods can achieve considerable performance improvement to promote the accuracy of CTR prediction to a new level. In future work, the methods of efficient GNN and lightweight graph construction will bring more application prospects to the application of graphs in the CTR field.

Acknowledgements. This work was partially supported by the Key Research Project of Zhejiang Province (No. 2022C01145) and the National Science Foundation of China (No. U20A20173 and No. 62125206).

References

Bo, D., Wang, X., Shi, C., Zhu, M., Lu, E. and Cui, P. (2020), Structural deep clustering network, *in* 'Proceedings of The Web Conference 2020', pp. 1400–1410.

Chen, H., Yin, H., Sun, X., Chen, T., Gabrys, B. and Musial, K. (2020), 'Multi-level graph convolutional networks for cross-platform anchor link prediction', *arXiv preprint arXiv:2006.01963* .

Cheng, H.-T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G., Chai, W., Ispir, M. et al. (2016), Wide & deep learning for recommender systems, *in* 'Proceedings of the 1st workshop on deep learning for recommender systems', pp. 7–10.

Cui, G., Zhou, J., Yang, C. and Liu, Z. (2020), Adaptive graph encoder for attributed graph embedding, *in* 'Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining', pp. 976–985.

Fan, W., Ma, Y., Li, Q., He, Y., Zhao, E., Tang, J. and Yin, D. (2019), Graph neural networks for social recommendation, *in* 'The World Wide Web Conference', pp. 417–426.

Fawcett, T. (2006), 'An introduction to roc analysis', *Pattern recognition letters* **27**(8), 861–874.

Guo, H., Tang, R., Ye, Y., Li, Z. and He, X. (2017), 'Deepfm: a factorization-machine based neural network for ctr prediction', *arXiv preprint arXiv:1703.04247* .

Hamilton, W., Ying, Z. and Leskovec, J. (2017), Inductive representation learning on large graphs, *in* 'Advances in neural information processing systems', pp. 1024–1034.

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

He, R. and McAuley, J. (2016), Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering, *in* 'proceedings of the 25th international conference on world wide web', pp. 507–517.

He, X., Deng, K., Wang, X., Li, Y., Zhang, Y. and Wang, M. (2020), 'Lightgcn: Simplifying and powering graph convolution network for recommendation', *arXiv preprint arXiv:2002.02126* .

Hjelm, R. D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Bachman, P., Trischler, A. and Bengio, Y. (2018), 'Learning deep representations by mutual information estimation and maximization', *arXiv preprint arXiv:1808.06670* .

Juan, Y., Zhuang, Y., Chin, W.-S. and Lin, C.-J. (2016), Field-aware factorization machines for ctr prediction, *in* 'Proceedings of the 10th ACM Conference on Recommender Systems', pp. 43–50.

Kim, K., Kwon, E. and Park, J. (2021), 'Deep user segment interest network modeling for click-through rate prediction of online advertising', *IEEE Access* **9**, 9812–9821.

Kipf, T. N. and Welling, M. (2016*a*), 'Semi-supervised classification with graph convolutional networks', *arXiv preprint arXiv:1609.02907* .

Kipf, T. N. and Welling, M. (2016*b*), 'Variational graph auto-encoders', *arXiv preprint arXiv:1611.07308* .

Li, D., Liu, H., Zhang, Z., Lin, K., Fang, S., Li, Z. and Xiong, N. N. (2022), 'Carm: Confidence-aware recommender model via review representation learning and historical rating behavior in the online platforms', *Neurocomputing* **455**, 283–296.

Li, F., Chen, Z., Wang, P., Ren, Y., Zhang, D. and Zhu, X. (2019), Graph intention network for click-through rate prediction in sponsored search, *in* 'Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval', pp. 961–964.

Li, X., Hu, Y., Sun, Y., Hu, J., Zhang, J. and Qu, M. (2020), 'A deep graph structured clustering network', *IEEE Access* .

Li, Z., Liu, H., Zhang, Z., Liu, T. and Xiong, N. N. (2021), 'Learning knowledge graph embedding with heterogeneous relation attention networks', *IEEE Transactions on Neural Networks and Learning Systems* .

Liu, H., Fang, S., Zhang, Z., Li, D., Lin, K. and Wang, J. (2021), 'Mfdnet: Collaborative poses perception and matrix fisher distribution for head pose estimation', *IEEE Transactions on Multimedia* .

Liu, H., Zheng, C., Li, D., Shen, X., Lin, K., Wang, J., Zhang, Z., Zhang, Z. and Xiong, N. N. (2020), 'Edmf: Efficient deep matrix factorization with review feature learning for industrial recommender system', *IEEE Transactions on Industrial Informatics* .

Liu, T., Liu, H., Li, Y.-F., Chen, Z., Zhang, Z. and Liu, S. (2019), 'Flexible ftir spectral imaging enhancement for industrial robot infrared vision sensing', *IEEE Transactions on Industrial Informatics* **16**(1), 544–554.

Liu, T., Liu, H., Li, Y., Zhang, Z. and Liu, S. (2018), 'Efficient blind signal reconstruction with wavelet transforms regularization for educational robot infrared vision sensing', *IEEE/ASME Transactions on Mechatronics* **24**(1), 384–394.

Mnih, A. and Kavukcuoglu, K. (2013), 'Learning word embeddings efficiently with noise-contrastive estimation', *Advances in neural information processing systems* **26**, 2265–2273.

Pironkov, G., Dupont, S. and Dutoit, T. (2016), Speaker-aware long short-term memory multi-task learning for speech recognition, *in* '2016 24th European Signal Processing Conference (EUSIPCO)', IEEE, pp. 1911–1915.

Qu, Y., Cai, H., Ren, K., Zhang, W., Yu, Y., Wen, Y. and Wang, J. (2016), Product-based neural networks for user response prediction, *in* '2016 IEEE 16th International Conference on Data Mining (ICDM)', IEEE, pp. 1149–1154.

Rendle, S. and Schmidt-Thieme, L. (2010), Pairwise interaction tensor factorization for personalized tag recommendation, *in* 'Proceedings of the third ACM international conference on Web search and data mining', pp. 81–90.

Richardson, M., Dominowska, E. and Ragno, R. (2007), Predicting clicks: estimating the click-through rate for new ads, *in* 'Proceedings of the 16th international conference on World Wide Web', pp. 521–530.

Shan, Y., Hoens, T. R., Jiao, J., Wang, H., Yu, D. and Mao, J. (2016), Deep crossing: Web-scale modeling without manually crafted combinatorial features, *in* 'Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining', pp. 255–262.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P. and Bengio, Y. (2017), 'Graph attention networks', *arXiv preprint arXiv:1710.10903* .

Veličković, P., Fedus, W., Hamilton, W. L., Liò, P., Bengio, Y. and Hjelm, R. D. (2018), 'Deep graph infomax', *arXiv preprint arXiv:1809.10341* .

Wang, X., He, X., Wang, M., Feng, F. and Chua, T.-S. (2019), Neural graph collaborative filtering, *in* 'Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval', pp. 165–174.

Wu, F., Zhang, T., Souza Jr, A. H. d., Fifty, C., Yu, T. and Weinberger, K. Q. (2019), 'Simplifying graph convolutional networks', *arXiv preprint arXiv:1902.07153* .

Xu, K., Hu, W., Leskovec, J. and Jegelka, S. (2018), 'How powerful are graph neural networks?', *arXiv preprint arXiv:1810.00826* .

Yan, L., Li, W.-J., Xue, G.-R. and Han, D. (2014), Coupled group lasso for web-scale ctr prediction in display advertising, *in* 'International Conference on Machine Learning', pp. 802–810.

Yi, B., Shen, X., Liu, H., Zhang, Z., Zhang, W., Liu, S. and Xiong, N. (2019), 'Deep matrix factorization with implicit feedback embedding for recommendation system', *IEEE Transactions on Industrial Informatics* **15**(8), 4591–4601.

Zhang, H., Ji, Y., Li, J. and Ye, Y. (2015), 'A triple wing harmonium model for movie recommendation', *IEEE Transactions on Industrial Informatics* **12**(1), 231–239.

Zhang, J., Shi, X., Zhao, S. and King, I. (2019), 'Star-gcn: Stacked and reconstructed graph convolutional networks for recommender systems', *arXiv preprint arXiv:1905.13129* .

Zhang, X., Wang, Z. and Du, B. (2021), 'Deep dynamic interest learning with session local and global consistency for click-through rate predictions', *IEEE Transactions on Industrial Informatics* .

Zhang, Z., Li, Z., Liu, H. and Xiong, N. N. (2020), 'Multi-scale dynamic convolutional network for knowledge graph embedding', *IEEE Transactions on Knowledge and Data Engineering* .

Zhao, W. X., Li, S., He, Y., Chang, E. Y., Wen, J.-R. and Li, X. (2015), 'Connecting social media to e-commerce: Cold-start product recommendation using microblogging information', *IEEE Transactions on Knowledge and Data Engineering* **28**(5), 1147–1159.

Zhou, G., Mou, N., Fan, Y., Pi, Q., Bian, W., Zhou, C., Zhu, X. and Gai, K. (2019), Deep interest evolution network for click-through rate prediction, *in* 'Proceedings of the AAAI conference on artificial intelligence', Vol. 33, pp. 5941–5948.

Zhou, G., Zhu, X., Song, C., Fan, Y., Zhu, H., Ma, X., Yan, Y., Jin, J., Li, H. and Gai, K. (2017), Deep interest network for click-through rate prediction, *in* 'Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining', pp. 1059–1068.

Zhou, J., Cui, G., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C. and Sun, M. (2018), 'Graph neural networks: A review of methods and applications', *arXiv preprint arXiv:1812.08434* .

Zhu, H., Jin, J., Tan, C., Pan, F., Zeng, Y., Li, H. and Gai, K. (2017), Optimized cost per click in taobao display advertising, *in* 'Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', pp. 2191–2200.

Author Biographies

Yixuan Wu was born in Handan, Hebei, China, in 1999. She received the B.S. degree in computer science from Shandong University, China, in 2021. She is currently pursuing the M.S. degree in computer science from Zhejiang University, China. Her research interests include recommender system and graph-related learning.

Youpeng Hu was born in Jiujiang, Jiangxi, China, in 1997. He received the B.S. degree in automation from Hangzhou Dianzi University, China, in 2019. He is currently pursuing the M.S. degree in computer science from Shandong University, China. His research interests include graph-related learning and intelligent information processing.

Xin Xiong was born in Huaihua, Hunan, China, in 1999. She received the B.S. degree in computer science from Shandong University, China, in 2021. She is currently pursuing the M.S. degree in computer science from Nanjing University, China. Her research interests include graph embedding.

Xunkai Li was born in Harbin, Heilongjiang, China, in 2000. He is currently pursuing the B.S. degree in computer science from Shandong University, China. His research interests include graph machine learning and federated learning.

Ronghui Guo was born in Ganzhou, Jiangxi, China, in 2000. He is currently pursuing the B.S. degree in computer science with Shandong University, China. His research interests include graph-related learning and intelligent information processing.

Shuiguang Deng is a full professor at the College of Computer Science and Technology in Zhejiang University. He received the BS and PhD both in Computer Science from Zhejiang University in 2002 and 2007, respectively. His research interests include Service Computing, Mobile Computing, and Edge Computing. Up to now he has published more than 100 papers in journals such as IEEE TOC, TPDS, TSC, TCYB and TNNLS, and refereed conferences. He is the Associate Editor of the journal IEEE Trans. on Services Computing and IET Cyber-Physical Systems: Theory & Applications. He is a senior member of IEEE.

Correspondence and offprint requests to: Shuiguang Deng, Zhejiang University, Hangzhou, China. Email: dengsg@zju.edu.cn

Representation Enhancement based Cold Start Model for Elastic Compute Service Recommendation

1st Yixuan Wu
Zhejiang University
Hangzhou, Zhejiang
yixuanwu@zju.edu.cn

2nd Xingliang Wang
Zhejiang University
Hangzhou, Zhejiang
wangxingliang@zju.edu.cn

3rd Siyu Deng
Alibaba
Hangzhou, Zhejiang
desy.dsy@alibaba-inc.com

4th Fei Peng
Alibaba
Hangzhou, Zhejiang
xinyou.pf@alibaba-inc.com

5th Zhengxiong Tian
Alibaba
Hangzhou, Zhejiang
zhengxiong.tianzx@alibaba-inc.com

6th Jian Zhou
Alibaba
Hangzhou, Zhejiang
zhoujian08@aliyun.com

7th Shuiguang Deng*
Zhejiang University
Hangzhou, Zhejiang
dengsg@zju.edu.cn

***Abstract*—As the core infrastructure service of cloud computing, Elastic Compute Service (ECS) offers a diverse range of products for users to select from. However, it is challenging for users to combine numerous configurations for the appropriate products, making it critical to be recommended individually. In contrast to traditional recommendation scenarios, ECS combines Business-to-Business (B2B) and Business-to-Consumer (B2C). Compared with enterprise users with rich behaviors, individual users account for a large proportion with limited behaviors, facing a serious cold start problem. Furthermore, enterprise users have high demands and strong preferences, leading to a high concentration on popular products, resulting in long tail distribution. To address the problems, we propose a novel model named the Representation Enhancement based Cold Start Model (ReCSM). Based on dynamic relations among users, the Cold start user Contextual Attention Module (CCAM) is designed to enhance the representation of cold start users with behavior-rich users, thus improving the expressiveness of the inductive model. What's more, the Generalized Attribute Clustering Embedding module (GACE) is proposed to maintain the clustering center representation for each user to improve the generalization ability. Additionally, we implement the Contrastive Learning based Long tail products Enhancement Module (CL-LEM) to relieve the effect of long tail distribution, ensuring the diversity and fairness of recommendation results. Extensive experiments have proved the validity of ReCSM on two real ECS datasets and one public dataset.**

***Index Terms*—Recommender System, Elastic Compute Service, Cold Start Problem, Long Tail Effect, Contrastive Learning**

I. Introduction

As the core service of cloud computing, Elastic Compute Service (ECS) is becoming increasingly important in a wide range of scenarios due to its stability, reliability, and scalability. With the number of products increasing, it becomes challenging for users to choose the appropriate products from numerous configurations. Therefore, it is crucial to recommend the proper products for users in the ECS selling domain, improving user experience and platform revenue, which are of significant research and commercial value.

* Corresponding author

It is the core step to model users' preferences by learning historical behavior data in the mainstream recommendation scenarios, such as service recommendation [1], [2], social network recommendation [3], [4], and e-commerce recommendation [5], [6], to filter out the corresponding candidate sets individually.

Compared with common recommendation scenarios, ECS combines Business-to-Business (B2B) and Business-to-Consumer (B2C). According to the number of their behavior, enterprise users with high demands and strong preferences are typically identified as warm start users in this scenario. In contrast, individual users with limited and diverse demands are typically identified as cold start users, while accounting for a large proportion. Therefore, ECS recommendation faces a more significant cold start problem. To address the cold start problem, side information including attribute [7]–[9] and context information [10], [11] is used to enrich user representation, but it struggles with limited expressiveness. The best strategy to improve the expressiveness of the model is to train ID embedding for each user to capture their behaviors and preferences. However, it is difficult to be trained sufficiently for users with a lack of behaviors, even impossible for new users, making the training process transductive. In summary, the key to solving the cold start problem is to balance expressiveness and generalization ability.

In addition, high demands and strong preferences from enterprise users lead to the dominance of popular products, which exacerbates the long tail effect. Thus, long tail products are challenging to train effectively due to limited behavior data. As a result, the model tends to recommend popular products, leading to a vicious circle that long tail products are less likely to be purchased due to few exposures. Inverse Propensity Weighting (IPW) [12], [13] is commonly used to alleviate the long tail effect, which accurately assigns higher weights to long tail products and lower weights to popular products.

In order to address the above problems in the ECS recommendation, we propose an inductive model named ReCSM, which improves the prediction accuracy of warm, cold, and strict cold start users, respectively. Additionally, long tail products are enhanced for better representation. The contributions of this paper are as follows:

- To enhance the representations of cold and strict cold start users, we propose the Cold start user Contextual Attention Module (CCAM) mingles adjacent warm users with rich behaviors, improving the expressive ability of the model.
- To strengthen the generalization of user features, we propose the Generalized Attribute Clustering Embedding module (GACE), which maintains the corresponding clustering center representation for each user as one of the model input features.
- To relieve the long tail effect on products, we propose the Contrastive Learning based Long tail products Enhancement Module (CL-LEM) for data augmentation on long tail products to incorporate unique information, ensuring the diversity and fairness of recommendation results.
- We demonstrate the effectiveness of ReCSM in ECS recommendation using two authentic datasets from Alibaba Cloud and prove its validity in common recommendation scenarios using the public dataset.

II. Preliminary

In this section, we introduce the main symbols and their explanations in Table I, and present the definitions of key concepts as follows:

Definition 1. Warm/Cold/Strict Cold Start Users. In this paper, we classify users into warm start users $\mathcal{U}_w$, cold start users $\mathcal{U}_c$, and strict cold start users $\mathcal{U}_{sc}$ based on the number of their purchase behaviors, where corresponding numbers are $N_w = |\mathcal{U}_w|$, $N_c = |\mathcal{U}_c|$, $N_{sc} = |\mathcal{U}_{sc}|$, for the total number $N = |\mathcal{U}|$. Warm start users have at least t behaviors, cold start users have fewer than t behaviors, and strict cold start users have not appeared in the training process. Warm start users are mostly enterprise users with abundant behaviors, whose preferences can be well captured. On the other hand, it is challenging to learn the preferences of cold and strict cold start users, which is the focus of this paper.

Definition 2. Long Tail Products. We specify products that have been purchased less than i as long tail products $\mathcal{I}_t$, whose number $M_t = |\mathcal{I}_t|$ and the total number of products $M = |\mathcal{I}|$.

Definition 3. Prediction Tasks for Diverse Users. For both warm and cold start users, ReCSM learns on the train data of corresponding users. However, for strict cold start users, ReCSM utilizes the train data of cold start users for prediction.

TABLE I: Symbols used in this paper

Symbols	Description
$\mathcal{U}_w$	Warm start users
$\mathcal{U}_c$	Cold start users
$\mathcal{U}_{sc}$	Strict cold start users
$N_w = \lvert\mathcal{U}_w\rvert$	The number of warm start users
$N_c = \lvert\mathcal{U}_c\rvert$	The number of cold start users
$N_{sc} = \lvert\mathcal{U}_{sc}\rvert$	The number of strict cold start users
$N = \lvert\mathcal{U}\rvert$	The total number of users
$\mathcal{I}_t$	Long tail products
$M_t = \lvert\mathcal{I}_t\rvert$	The number of long tail products
$M = \lvert\mathcal{I}\rvert$	The total number of products
$\mathbf{pre}_w$	The pretrained ID embedding of warm start users
$\mathbf{Tr}_w$	The train data of warm start users
$\mathbf{Te}_w$	The test data of warm start users
$\mathbf{Tr}_c$	The train data of cold start users
$\mathbf{Te}_c$	The test data of cold start users
$\mathbf{Te}_{sc}$	The test data of strict cold start users

III. Method

A. Model Framework

Each sample of purchasing behavior can be represented as $(\mathbf{x}, y)$, where $y \in \{0, 1\}$ denotes whether the user purchased the target product, $\mathbf{x} = \left[\mathbf{x}_{\mathcal{U}}^{id}, \mathbf{x}_{\mathcal{U}}^{att}, \mathbf{x}_{\mathcal{U}}^{clu}, \mathbf{x}_{\mathcal{I}}^{id}, \mathbf{x}_{\mathcal{I}}^{att}, ...\right]$ represents the features of user id, user attribute, user cluster, product id, product attribute, or others.

Since discrete features are often encoded to sparse one-hot vectors, they are densified through an embedding layer:

$$\mathbf{e}_i = f_i\left(\mathbf{x}_i\right), \tag{1}$$

where $\mathbf{x}_i$ represents the above feature, and $\mathbf{e}_i$ represents the corresponding embedding. f_i briefly follows the table lookup mechanism in general, while $f_{\mathcal{U}}^{id}$, $f_{\mathcal{U}}^{clu}$ and $f_{\mathcal{I}}^{id}$ are optimized in section III-B, section III-C and section III-D, respectively. The specific method of optimization is shown in Figure 1.

Furthermore, the prediction result is obtained through feature interaction:

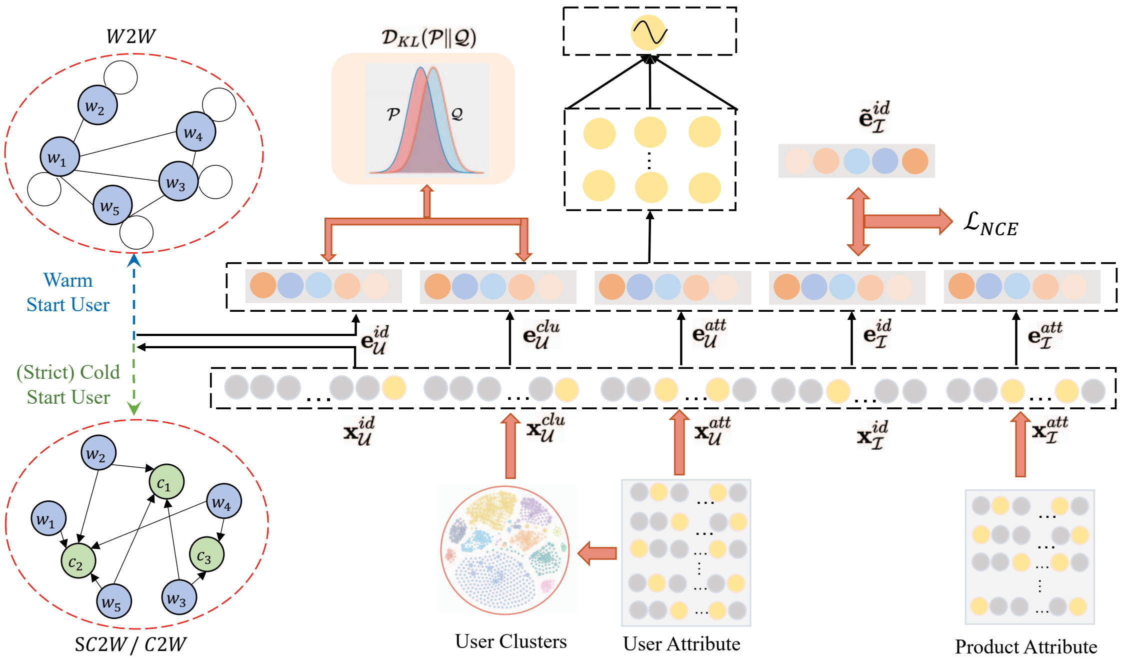
$$\hat{y} = \sigma\left(g\left(\mathbf{e}_{\mathcal{U}}^{id}, \mathbf{e}_{\mathcal{U}}^{att}, \mathbf{e}_{\mathcal{U}}^{clu}, \mathbf{e}_{\mathcal{I}}^{id}, \mathbf{e}_{\mathcal{I}}^{att}, ...\right)\right), \tag{2}$$

where $g(\cdot)$ represents the feature interaction function, multi-layer perceptrons are used here followed [14], $\sigma(t) = \frac{1}{1+e^{-t}}$ denotes the sigmoid activation function, and $\hat{y}$ is the prediction result of the current data $\mathbf{x}$.

In the training process, the binary cross entropy loss function is utilized to calculate the loss of the prediction:

$$\mathcal{L}_{BCE} = -\frac{1}{n}\sum_{j}^{n}\left[y_j \log \hat{y}_j + (1 - y_j)\log\left(1 - \hat{y}_j\right)\right]. \tag{3}$$

To optimize the training process, we design the auxiliary tasks in section III-C and section III-D, whose loss functions are added to $\mathcal{L}_{BCE}$ as penalty terms for joint training.



$W2W$
$\mathcal{D}_{KL}(\mathcal{P}\|\mathcal{Q})$
$\tilde{\mathbf{e}}_{\mathcal{I}}^{id}$
$\mathcal{L}_{NCE}$
Warm Start User
(Strict) Cold Start User
$\mathbf{e}_{\mathcal{U}}^{id}$
$\mathbf{e}_{\mathcal{U}}^{clu}$
$\mathbf{e}_{\mathcal{U}}^{att}$
$\mathbf{e}_{\mathcal{I}}^{id}$
$\mathbf{e}_{\mathcal{I}}^{att}$
$\mathbf{x}_{\mathcal{U}}^{id}$
$\mathbf{x}_{\mathcal{U}}^{clu}$
$\mathbf{x}_{\mathcal{U}}^{att}$
$\mathbf{x}_{\mathcal{I}}^{id}$
$\mathbf{x}_{\mathcal{I}}^{att}$
$SC2W$ / $C2W$
User Clusters
User Attribute
Product Attribute

Fig. 1: The overall framework of ReCSM, where $\mathbf{e}_{\mathcal{U}}^{id}$, $\mathbf{e}_{\mathcal{U}}^{clu}$, and $\mathbf{e}_{\mathcal{I}}^{id}$ are enhanced by CCAM, GACE, and CL-LEM, respectively

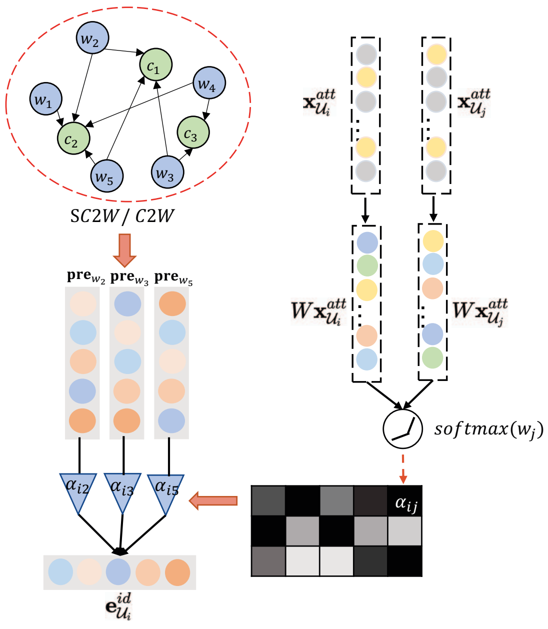
B. Cold Start Contextual Attention Module

It is essential to model users' preferences through their historical behavior as ID embeddings. However, training ID embeddings for cold and strict cold start users is challenging due to insufficient behavioral data. Therefore, we propose CCAM to enhance the representations of cold and strict cold start users with the rich data of warm start users. The specific module is shown in Figure 2.

Firstly, we pre-train an ID embedding $\mathbf{pre}_w \in R^d$ using rich behaviors of warm start users to accurately reflect their preferences. Specifically, we pre-train according to section III-A, where features include user id, user attribute, user cluster, product id, and product attribute. f_i follows the table lookup mechanism, except $f_{\mathcal{U}}^{clu}$ is optimized in section III-C. The preference representation of warm start users $\mathbf{pre}_w$ is continuously optimized during the pre-training process.

Secondly, we calculate the attribute similarity between cold and warm start users to construct the graph C2W, where C2W $\in R^{N_c \times N_w}$. C2W is a bipartite graph with nodes representing cold and warm start users respectively. An edge is created between the cold and the warm start user if their attribute similarity exceeds the threshold D. SC2W $\in R^{N_{sc} \times N_w}$ is constructed in the same way by calculating the similarity between strict cold and warm start users.

In order to accurately measure the contribution of different adjacent warm start users $w_j \in \mathcal{N}_{c_i}$ to the cold start users c_i, we learn the adaptive coefficient $\alpha_{c_i}^{w_j}$ to aggregate the well-



$SC2W$ / $C2W$
$\mathbf{x}_{\mathcal{U}_i}^{att}$
$\mathbf{x}_{\mathcal{U}_j}^{att}$
$W\mathbf{x}_{\mathcal{U}_i}^{att}$
$W\mathbf{x}_{\mathcal{U}_j}^{att}$
$\mathbf{pre}_{w_2}$ $\mathbf{pre}_{w_3}$ $\mathbf{pre}_{w_5}$
$softmax(w_j)$
α_{i2}
α_{i3}
α_{i5}
α_{ij}
$\mathbf{e}_{\mathcal{U}_i}^{id}$

Fig. 2: The specific implementation of CCAM

trained $\mathbf{pre}_w$ to enrich the information of cold start users:

$$\alpha_{c_i}^{w_j} = \frac{\exp\left(\text{LeakyReLU}\left(\left[\mathbf{W}\mathbf{x}_{c_i}^{att} \| \mathbf{W}\mathbf{x}_{w_j}^{att}\right]\right)\right)}{\sum_{w_k \in \mathcal{N}_{c_i}} \exp\left(\text{LeakyReLU}\left(\left[\mathbf{W}\mathbf{x}_{c_i}^{att} \| \mathbf{W}\mathbf{x}_{w_k}^{att}\right]\right)\right)}, \tag{4}$$

where $\mathbf{x}_{c_i}^{att} \in R^d$ and $\mathbf{x}_{w_j}^{att} \in R^d$ represent the corresponding attribute features, d represents its dimension, and $\mathbf{W} \in R^{d\times d}$ is the weight matrix. By aggregating the well-trained ID embeddings of neighbor warm start users, the self-node representation is enhanced as:

$$\mathbf{e}_{c_i}^{id} = \sigma\left(\sum_{w_j \in \mathcal{N}_{c_i}} \alpha_{c_i}^{w_j} \mathbf{W}\mathbf{pre}_{w_j}\right). \tag{5}$$

Similarly, every strict cold start user can generate its preference representation $\mathbf{e}_{sc_i}^{id}$ as above.

What's more, for warm start users, we further optimize their representations based on $\mathbf{pre}_w$. To provide warm start users with more comprehensive information, we construct W2W $\in R^{N_w \times N_w}$ with nodes representing warm start users. An edge is created between every two users if their attribute similarity exceeds the threshold D. We apply the adaptive coefficients to aggregate the neighbor nodes on W2W for each warm start user, where the coefficient of the two warm start users w_i and w_j is:

$$\alpha_{w_i}^{w_j} = \frac{\exp\left(\text{LeakyReLU}\left(\left[\mathbf{W}\mathbf{x}_{w_i}^{att} \| \mathbf{W}\mathbf{x}_{w_j}^{att}\right]\right)\right)}{\sum_{w_k \in \mathcal{N}_{c_i}} \exp\left(\text{LeakyReLU}\left(\left[\mathbf{W}\mathbf{x}_{w_i}^{att} \| \mathbf{W}\mathbf{x}_{w_k}^{att}\right]\right)\right)}. \tag{6}$$

We aggregate neighbors adaptively to effectively enhance the representation of warm start user w_i:

$$\mathbf{e'}_{w_i}^{id} = \sigma\left(\sum_{w_j \in \mathcal{N}_{c_i}} \alpha_{w_i}^{w_j} \mathbf{W}\mathbf{pre}_{w_j}\right). \tag{7}$$

Finally, the node representation is updated as its preference:

$$\mathbf{e}_{w_i}^{id} = Relu(\mathbf{W}'([\mathbf{e'}_{w_i}^{id} \| \mathbf{pre}_{w_i}])), \tag{8}$$

where the weight matrix $\mathbf{W}' \in R^{2d\times d}$.

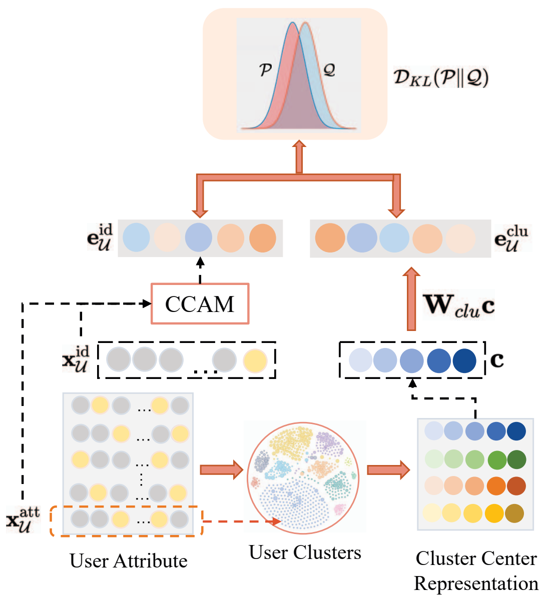
C. Generalized Attribute Clustering Embedding module

We design a graph clustering task on user attributes, named GACE shown in Figure 3, to identify the cluster center representation $\mathbf{x}_{\mathcal{U}}^{clu}$ for each user as the input feature of the model. By jointly maintaining the cluster center representation, both cold and strict cold start users can be learned alongside other behavior-rich users in the same cluster, improving the generalization ability of the model. Additionally, GACE standardizes the representations of users and their corresponding clusters, which improves the cohesiveness of user representations within the same cluster, making it more suitable for downstream tasks.

Specifically, we use the classical k-means algorithm [15] to cluster users based on their attribute features $\mathbf{X}_{\mathcal{U}}^{att} \in R^{N\times d}$, generating the initial parameters $\mathbf{C} \in R^{K\times d}$ of K cluster center representations and determining the cluster to which each user belongs as $\mathcal{U}_i \in \mathbf{c}$.

In addition, we learn the cluster center representations as the input feature of the model by updating the weight matrix $\mathbf{W_{clu}} \in R^{d\times d}$ in the process of training:

$$\mathbf{e}_{\mathcal{U}_i}^{clu} = \mathbf{W_{clu}}\mathbf{c}^T, \tag{9}$$



$\mathcal{P}$
$\mathcal{Q}$
$\mathcal{D}_{KL}(\mathcal{P}\|\mathcal{Q})$
$\mathbf{e}_{\mathcal{U}}^{\text{id}}$
$\mathbf{e}_{\mathcal{U}}^{\text{clu}}$
CCAM
$\mathbf{W}_{clu}\mathbf{c}$
$\mathbf{x}_{\mathcal{U}}^{\text{id}}$
$\mathbf{c}$
$\mathbf{x}_{\mathcal{U}}^{\text{att}}$
User Attribute
User Clusters
Cluster Center Representation

Fig. 3: An illustration of GACE

where the cluster center embedding is maintained jointly by users in the same cluster.

What's more, referring to DAEGC [16], we calculate the similarity of users and their clustering centers as the probability that a user belongs to this cluster, i.e., the distribution of users:

$$q_{uc} = \frac{\left(1 + \left\|\mathbf{e}_{\mathcal{U}_i}^{id} - \mathbf{e}_{\mathcal{U}_i}^{clu}\right\|^2\right)^{-1}}{\sum_k \left(1 + \left\|\mathbf{e}_{\mathcal{U}_i}^{id} - \mathbf{e}_{\mathcal{U}_k}^{clu}\right\|^2\right)^{-1}}. \tag{10}$$

The target distribution of users and corresponding clustering centers is:

$$p_{uc} = \frac{q_{uc}^2 / \sum_u q_{uc}}{\sum_k \left(q_{uk}^2 / \sum_u q_{uk}\right)}. \tag{11}$$

We force the distribution Q to be closer to the target distribution P by minimizing Kullback-Leibler Divergence [17] of the two distributions:

$$\mathcal{L}_{KL} = KL(P\|Q) = \sum_u \sum_c p_{uc} \log \frac{p_{uc}}{q_{uc}}. \tag{12}$$

In summary, we minimize $\mathcal{L}_{KL}$ by decreasing the intra-class distance and increasing the inter-class distance, thus normalizing the embedding $\mathbf{e}_{\mathcal{U}}^{id}$ of the user and the embedding $\mathbf{e}_{\mathcal{U}}^{clu}$ of the cluster to which the user belongs.

D. Contrastive Learning based Long tail Products Enhancement Module

It is challenging for long tail products to be learned effectively due to limited training data. Thus we propose CL-LEM which employs contrastive learning to augment the representations of long tail products, thereby mitigating the

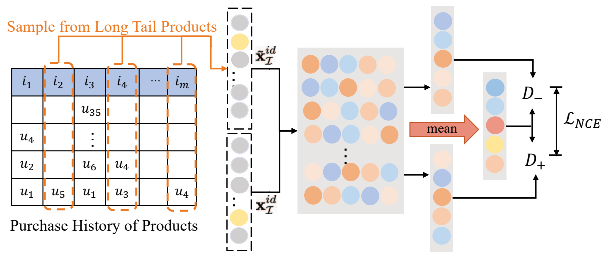

Sample from Long Tail Products
i_1 i_2 i_3 i_4 $\cdots$ i_m
u_{35}
u_4
u_2 u_6 u_4
u_1 u_5 u_1 u_3 u_4
Purchase History of Products
$\tilde{\mathbf{x}}_{\mathcal{I}}^{id}$
$\mathbf{x}_{\mathcal{I}}^{id}$
mean
D_-
D_+
$\mathcal{L}_{NCE}$

Fig. 4: The details of CL-LEM

impact of the long tail distribution and ensuring the diversity and fairness of recommendation results, shown in Figure 4.

The core idea of contrastive learning is to find three components from the original data: positive sample, negative sample, and anchor.

For each sample $(\mathbf{x}_{\mathcal{U}_i}, \mathbf{x}_{\mathcal{I}_j}, \ldots, y)$, we randomly sample a negative sample $\widetilde{\mathcal{I}}_t$ from long tail products that the user $\mathcal{U}_i$ has not purchased. We then utilize the same parameter-sharing embedding layer, which is followed by the lookup mechanism, to obtain representations $\mathbf{e}_{\mathcal{I}_j}$ and $\widetilde{\mathbf{e}}_{\mathcal{I}_t}$ for the positive sample $\mathcal{I}_j$ and the negative sample $\widetilde{\mathcal{I}}_t$ respectively. In addition, the mean function is used as the readout step to extract the global product representation as the anchor:

$$\mathbf{R} = \sigma\left(\frac{1}{M}\sum_{j=1}^{M}\mathbf{e}_{\mathcal{I}_j}\right). \tag{13}$$

Furthermore, a bilinear scoring function is utilized as the discriminator:

$$D\left(\mathbf{e}_{\mathcal{I}_j}, \mathbf{R}\right) = \sigma\left(\mathbf{e}_{\mathcal{I}_j}\mathbf{W}_{\mathbf{con}}\mathbf{R}\right), \tag{14}$$

where $\mathbf{W}_{\mathbf{con}} \in R^{d\times d}$ is a learnable scoring matrix to generate the score of sample for $\left(\mathbf{e}_{\mathcal{I}_j}, \mathbf{R}\right)$. Finally, we use contrastive noise estimation (NCE) loss [18] to maximize the mutual information between local representations and global representations, i.e., the JS divergence between the joint distribution and the product of marginal distribution:

$$\begin{aligned}\mathcal{L}_{NCE} = -\frac{1}{2M}\Big(&\sum\mathbb{E}\left[\log D\left(\mathbf{e}_{\mathcal{I}_j}, \mathbf{R}\right)\right] \\ &+\sum\mathbb{E}\left[1-\log D\left(\tilde{\mathbf{e}}_{\mathcal{I}_t}, \mathbf{R}\right)\right]\Big).\end{aligned} \tag{15}$$

By minimizing $\mathcal{L}_{NCE}$, the representations of long tail products are enriched with specific and unique global information without extra labels, relieving the long tail effect.

In summary, we use self-supervised clustering and contrastive learning as auxiliary tasks to optimize the main task prediction for end-to-end training:

$$\mathcal{L} = \mathcal{L}_{BCE} + \alpha\mathcal{L}_{KL} + \beta\mathcal{L}_{NCE}, \tag{16}$$

where α is the weight of the clustering auxiliary task and β is the weight of the contrastive auxiliary task. It is worth mentioning that the multi-task learning paradigm improves the robustness of the model without any computational burden during inference.

TABLE II: Statistics of the datasets

Dataset	User	Item	Samples	Density	Samples Per User
ECS-QL	71,842	747	1,947,634	3.63%	27
ECS-CL	540,605	1,536	36,054,866	4.34%	67
ML-100k	943	1,682	100,000	6.30%	106

IV. Experiments

A. Experimental Settings

1) Datasets: The statistical information of datasets is shown in Table II, and the description is as follows:

- **ECS Datasets** includes ECS Quick Launch (ECS-QL) and ECS Custom Launch (ECS-CL), which are partly selected from two different ECS sales modes from Alibaba Cloud. We generate the same number of negative samples as positive samples for each user from products that were exposed to users but not purchased. Our goal is to predict each user's n-th behavior based on the past $n-1$ behaviors.
- **MovieLens** [1] is chosen to verify the effectiveness of our model in common recommendation scenarios. To align with our task, we convert the rating prediction to a binary classification task by classifying user ratings greater than 3 as positive and others as negative samples. We use 90% of each user's behavior data as the train set and 10% as the test set.

2) Datasets Analysis: We conduct statistical analysis of the cold start users and long tail items on the three datasets. The specific information is shown in Figure 5, which consists of three nested pies, each corresponding to a dataset.

The inner pie displays the proportion of warm, cold, and strict cold start users, while the second pie shows the proportion of the number of samples per user type. Although users with less than 20 ratings have been removed from ML-100k, resulting in no strict cold start users, a proportion of users still face the cold start problem. Both ECS datasets show a high proportion of cold and strict cold start users, with cold start users accounting for over half and strict cold start users accounting for over a quarter. This highlights significant cold start challenges in ECS compared to typical recommendation scenarios. However, the small number of their corresponding samples makes it difficult to train effectively based solely on their interaction data.

Furthermore, we classify long tail items into four types, based on the number of times they were purchased (PN $\leq$ 1, 2, 4, and 8). The proportion of each type is displayed in the outer pie. ML-100k has an average of 59 samples per item, with a certain proportion of items having fewer samples. The two ECS datasets have a large number of samples per item, with an average of 2,607 and 23,473 respectively. However, more than one-quarter of the items have a purchase frequency of no more than 8, indicating a significant long tail distribution

[1]https://files.grouplens.org/datasets/movielens/ml-100k/

in ECS recommendation. This means that a small number of head items are purchased frequently, while the majority of tail items are purchased infrequently.

Overall, data analysis has proven that solving the cold start problem and the long tail effect is crucial for ECS recommendation and can also improve the performance of common recommendations.

3) Baselines: We verify our model on 3 transductive models and 3 inductive models.

- **NNMF** [19]: A transductive model uses neural networks to decompose the co-occurrence matrix of users and items for the hidden vector representations.
- **GCMC** [20]: A transductive model introduces a graph encoder to complement high-level information for users and items.
- **DeepFM** [20]: A transductive model extracts both low- and high-order feature interactions through a combination of FM and MLP.
- **GAT** [21]: An inductive model enables specifying different weights to different nodes in a neighborhood for better node representation.
- **IDCF** [22]: An inductive model computes embeddings for query users from key users via attention-based structure on historical behaviors.
- **AGNN** [23]: An inductive model generates preference representations for cold start users or items by learning the attribute distributions.

4) Matrics: In the binary classification task, AUC is widely used to evaluate the effectiveness of models through the ability to distinguish between positive and negative samples. Furthermore, RelaImpr is introduced to measure relative improvement over models followed [14]:

$$RelaImpr = (\frac{AUC(objective\ model) - 0.5}{AUC(base\ model) - 0.5} - 1) \times 100\%. \tag{17}$$

5) Implementation: To verify the validity, the hyperparameters for all models are consistent. All models are learned by the Adam optimizer, the learning rate is set to 0.001, and its decay rate is 0.9 per 336,000 samples. For all datasets, the training batch size is set to 64, the testing batch size is set to 32, and the embedding dimension d is set to 128.

In CCAM, the threshold t for the number of behaviors required for warm start users $\mathcal{U}_w$ is set to 10 for ECS-QL and ECS-CL, and to 30 for the MovieLens dataset. In GACE, the number of clusters K is set to 15 for ECS-QL and ECS-CL, and 4 for MovieLens. Additionally, α is set to 1 for ECS-QL and ECS-CL, and 0.01 for MovieLens. In CL-LEM, for the three datasets, we set the threshold i for the number of purchased times for long tail products $\mathcal{I}_t$ to 2. Moreover, β is set to 0.5 for ECS-QL and ECS-CL, and 0.6 for MovieLens. For downstream feature interaction, we apply a three-layer perceptron with 200, 80, and 1 neurons, respectively. While ReLU is used as the activation function for all layers except the last one.

TABLE III: Results of the comparative study on the three datasets, where the bolded numbers in the table represent the best results and the underlined numbers represent the suboptimal results

Models	Params	Strict Cold		Cold		Warm	
		AUC	RelaImpr	AUC	RelaImpr	AUC	RelaImpr
ECS-QL Dataset							
NNCF	23.44M	0.9002	0.00%	0.8798	0.00%	0.9245	0.00%
GCMC	24.46M	0.9043	1.02%	0.8806	0.21%	0.8986	-6.10%
DeepFM	23.54M	0.8723	-6.97%	0.8814	0.42%	$\underline{0.9347}$	$\underline{2.40\%}$
GAT	9.46M	0.9131	3.22%	0.8867	1.82%	0.8747	-11.73%
IDCF	23.74M	-	-	0.8904	2.79%	0.8634	-14.39%
AGNN	0.25M	$\underline{0.9196}$	$\underline{4.85\%}$	$\underline{0.8938}$	$\underline{3.69\%}$	0.8824	-9.92%
ReCSM	10.01M	**0.9233**	**5.77%**	**0.9038**	**6.32%**	**0.9455**	**4.95%**
ECS-CL Dataset							
NNCF	173.69M	0.9147	0.00%	0.9048	0.00%	0.8780	0.00%
GCMC	174.72M	0.9191	1.06%	0.9107	1.46%	0.8973	5.11%
DeepFM	173.74M	0.8992	-3.74%	0.8949	-2.45%	0.8950	4.50%
GAT	69.59M	0.9189	1.01%	0.9103	1.36%	0.8787	0.19%
IDCF	174.00M	-	-	0.9065	0.42%	0.8890	2.91%
AGNN	17.97M	$\underline{0.9234}$	$\underline{2.10\%}$	$\underline{0.9133}$	$\underline{2.10\%}$	$\underline{0.9027}$	$\underline{6.53\%}$
ReCSM	69.78M	**0.9319**	**4.15%**	**0.9241**	**4.77%**	**0.9122**	**9.05%**
MovieLens Dataset							
NNCF	92.39K	0.6286	0.00%	0.6263	0.00%	0.7406	0.00%
GCMC	96.55K	$\underline{0.6694}$	$\underline{31.73\%}$	0.6444	14.33%	0.7655	10.35%
DeepFM	91.13K	0.6285	-0.08%	0.6115	-11.72%	0.7283	-5.11%
GAT	359.07K	0.6479	15.01%	0.6688	33.65%	0.7442	1.50%
IDCF	96.54K	0.6488	15.71%	0.6250	-1.03%	$\underline{0.7670}$	$\underline{10.97\%}$
AGNN	123.68K	0.6172	-8.87%	$\underline{0.6777}$	$\underline{40.70\%}$	0.7554	6.15%
ReCSM	334.75K	**0.6791**	**39.27%**	**0.7245**	**77.75%**	**0.7764**	**14.88%**

B. Comparative Study

We validate the effectiveness of our model on ECS-QL, ECS-CL, and MovieLens datasets based on three transductive models and three inductive models, respectively. The experiment results show in table III, where IDCF requires the users' behavior history to calculate similarity, making it unable to handle the strict cold start problem.

It shows that transductive models typically perform better on warm start users, but worse on cold and strict cold users. Because they can model warm start users with rich data, but struggle to represent cold start users with less behavior and are unable to handle strict cold start users not appearing in the training set. Inductive models are more effective in prediction for cold and strict cold start users than transductive models. They utilize attribute features or correlative users as side information to improve the generalization of models for

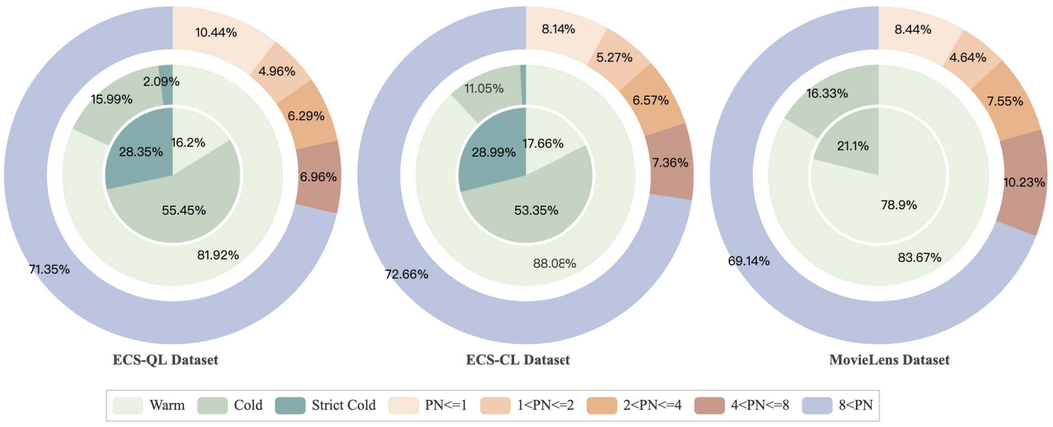


Fig. 5: The analysis results of the three datasets. The inner pie displays the proportion of warm, cold, and strict cold start users, while the second pie shows the proportion of samples per user type. Additionally, the outer pie represents the distribution of different types of items based on the number of purchased times (PN).

cold start users. However, the models struggle to accurately capture the personalized demands of warm start users, thus reducing the upper limit of expressive ability. In contrast, our model employs different data augmentation for cold start and warm start users, thus adapting to all types of users. Furthermore, our model has exceptional real-time performance due to its minimal number of trainable parameters. With enhanced representations of users and products, it can perform well on downstream tasks with few layers. Additionally, we observed that ReCSM predicts more accurately for strict cold start users than cold start users in ECS-QL and ECS-CL datasets. It reveals that the training data of cold start users can fit the first purchase behavior of strict cold start users well, and also justifies our training on $\mathbf{Tr}_c$ to predict $\mathbf{Te}_{sc}$.

Unlike ECS datasets with behavior-rich and preference-focused enterprise users, the MovieLens dataset only contains individual users, resulting in a lower density of data to hardly capture users' preferences. As a result, the AUC on MovieLens dataset is generally lower than ECS datasets.

C. Ablation Study

We conduct ablation studies on three datasets to demonstrate the impact of each module. Specifically, we remove the pre-trained module, CCAM, GACE, and CL-LEM individually from ReCSM. The experimental results are shown in the table IV. It can be seen that CCAM is the most important part of ReCSM on the two ECS datasets. As the key to solving the cold start problem, CCAM enhances the preference representation of cold start users through warm start users with rich data, improving the expressive ability of the inductive model. However, it is crucial for ReCSM to pre-train on the MovieLens dataset, which can alleviate the data sparsity problem by rich behavior data.

D. Cold Start Study

In the cold start scenario, we leverage information from warm start users to enhance the representation of cold and strict cold start users. It means that a smaller proportion of warm start users results in less side information to enhance the representation of cold and strict cold start users. Therefore, we verify the robustness of ReCSM by reducing the proportion of

TABLE IV: Results of the ablation study on the three datasets, where the bolded numbers in the table represent the best results and the numbers with ↓ are the least effective

Variations	Strict Cold		Cold		Warm	
	AUC	RelaImpr	AUC	RelaImpr	AUC	RelaImpr
ECS-QL Dataset						
ReCSM-Pretrain	0.9226	-0.17%	0.9019	-0.47%	0.9411	-0.99%
ReCSM-CCAM	0.9175	-1.37% ↓	0.9033	-0.12%	0.9444	-0.25%
ReCSM-GACE	0.9209	-0.57%	0.9031	-0.17%	0.9330	-2.81% ↓
ReCSM-CL-LEM	0.9222	-0.26%	0.9018	-0.5% ↓	0.9455	0.0%
ReCSM	**0.9233**	0.00%	**0.9038**	0.00%	**0.9455**	0.00%
ECS-CL Dataset						
ReCSM-Pretrain	0.9312	-0.16%	0.9233	-0.19%	0.9034	-2.13%
ReCSM-CCAM	0.9286	-0.76% ↓	0.9215	-0.61% ↓	0.9054	-6.10% ↓
ReCSM-GACE	0.9305	-0.32%	0.9218	-0.54%	0.9067	-1.33%
ReCSM-CL-LEM	0.9311	-0.19%	0.9234	-0.17%	0.9072	-1.21%
ReCSM	**0.9319**	0.00%	**0.9241**	0.00%	**0.9122**	0.00%
MovieLens Dataset						
ReCSM-Pretrain	0.6586	-11.45% ↓	0.6965	-12.47% ↓	0.7755	-0.33%
ReCSM-CCAM	**0.6948**	8.77%	0.7094	-6.73%	0.7682	-2.97% ↓
ReCSM-GACE	0.6725	-3.69%	0.7051	-8.64%	**0.7781**	0.62%
ReCSM-CL-LEM	0.6753	-2.12%	0.7199	-2.05%	0.7755	-0.33%
ReCSM	0.6791	0.00%	**0.7245**	0.00%	0.7764	0.00%

TABLE V: AUC of the cold start study on the three datasets, by reducing the proportion of warm start users

Ratio	80%	60%	40%	20%	10%
ECS-QL Dataset					
Strict Cold	0.9225	0.9234	0.9196	0.922	0.9222
Cold	0.9025	0.9028	0.9028	0.9011	0.9022
ECS-CL Dataset					
Strict Cold	0.9302	0.9306	0.9304	0.9305	0.9304
Cold	0.9227	0.9241	0.9230	0.9233	0.9237
MovieLens Dataset					
Strict Cold	0.6761	0.6801	0.6817	0.6743	0.6666
Cold	0.7166	0.7011	0.7128	0.7100	0.7027

TABLE VI: Results of the long tail study on the ECS-QL dataset

Purchased Number	≤1	≤2	≤4	≤8
Number of Products	78	115	162	214
AUC	0.9536	0.8959	0.9233	0.9257

warm start users. We randomly reduce the proportion of warm start users to 80%, 60%, 40%, 20%, and 10% of original data, respectively.

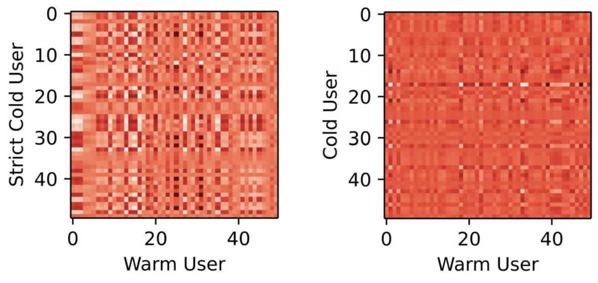
The results indicate that ReCSM maintains high prediction accuracy for both cold and strict cold start users, in Table V. This shows that ReCSM effectively mines information from data and learns accurate representations for cold start users with minimal data. Its immunity to the number of warm start users makes it ideal for situations with a significant number of cold start users.

E. Long Tail Study

To verify that ReCSM can alleviate the long tail effect, we predict samples of long tail products that have been purchased no more than 1, 2, 4, and 8. The results show in Table VI that ReCSM can achieve excellent performance even for long tail products, thus demonstrating that CL-LEM can learn sufficiently specific representations by data augmentation of long tail products. As the number of behaviors increases, products can be more thoroughly learned, making it easier to capture purchased tendency. However, products with unique behavior deviate from this trend, as users who purchase these unusual products have strong preferences that are prone to be predicted.

F. Visualization Study

We conduct the visualization study on the contribution of warm start users to cold and strict cold start users by randomly selecting 50 users from each group. It can be seen that a subset of warm start users make a significant contribution to all cold start users in Figure 6a and Figure 6b. Therefore, we can reasonably speculate that a small number of important users can play a key role in enriching the representation of cold start users, as demonstrated by the experiment results of the cold start study in section IV-D.



(a) Warm / Strict Cold User (b) Warm / Cold User

Fig. 6: The attention values between different types of users

G. A/B Testing

We conducted A/B testing from 2022-11 to 2022-12 in Alibaba Cloud comparing ReCSM with the last version of our online-serving model. The experiment results show that ReCSM contributes up to 20.78% CTR and 19.83% CVR (UV order rate), indicating the significant effectiveness and industrial value of our model.

V. Related Work

A. Cold Start Recommendation

The common solution to the cold start problem is using side information, such as user attributes, item attributes, contextual information, and cross-domain information of users and items. [23] designs a variant of graph neural network named Gated-GNN, to efficiently aggregate side information from different modalities in neighboring nodes. An extended variational auto-encoder (eVAE) [24] module is also introduced to learn the distribution of attributes and generate preference representations for strict cold start users and items. IDCF [22] resorts to attention-based structure learning that estimates hidden relations from query to key users and learns to leverage meta latents to inductively compute embeddings for query users via neural message passing.

B. Long Tail Effect

To mitigate the long tail effect, recommendation systems mainly focus on addressing bias. Inverse Propensity Weighting (IPW) [25] is the main method to debias, which assigns higher weights to long tail items and lower weights to popular items. Another strategy utilizes neural networks to model the biased values separately [26]–[28]. For example, [29] mitigates the selection biases by adopting a Wide & Deep framework.

VI. Conclusion

In this paper, we propose a novel ECS recommendation model named ReCSM, which aims to improve user purchase experience and platform revenue.

To enhance the representation of cold start users, CCAM is proposed to aggregate users with rich behaviors using adaptive coefficients. Additionally, we design GACE to maintain the corresponding clustering center representation for each user. Furthermore, CL-LEM is introduced for data augmentation on long tail products to ensure diversity and fairness of recommendation results. We conduct extensive experiments on two real ECS datasets and the results verify that ReCSM outperforms existing transductive and inductive models. What's more, we prove that ReCSM can be well generalized to common recommendation scenarios through the public dataset.

In future work, the methods of lightweight graph construction will improve the performance of real-time recommendations.

Acknowledgment

This work was supported in part by the Key Research Project of Zhejiang Province under Grant 2022C01145 and in part by the National Science Foundation of China under Grants U20A20173 and 62125206.

References

[1] M. Liu, D. Yu, Z. Tu, and Z. Wang, "srvpa: A multi-domain conversational service recommendation approach," in *2022 IEEE International Conference on Web Services (ICWS)*, pp. 170–175, IEEE, 2022.

[2] C. Wei, Y. Fan, J. Zhang, and H. Lin, "A-hsg: Neural attentive service recommendation based on high-order social graph," in *2020 IEEE International Conference on Web Services (ICWS)*, pp. 338–346, IEEE, 2020.

[3] D. Wang, X. Wang, Z. Xiang, D. Yu, S. Deng, and G. Xu, "Attentive sequential model based on graph neural network for next poi recommendation," *World Wide Web*, vol. 24, no. 6, pp. 2161–2184, 2021.

[4] H. Chen, Z. Feng, S. Chen, X. Xue, H. Wu, Y. Sun, Y. Xu, and G. Han, "Capturing users' fresh interests via evolving session-based social recommendation," in *2022 IEEE International Conference on Web Services (ICWS)*, pp. 182–187, IEEE, 2022.

[5] H. Ji, J. Zhu, C. Shi, X. Wang, B. Wang, C. Zhang, Z. Zhu, F. Zhang, and Y. Li, "Large-scale comb-k recommendation," in *Proceedings of the Web Conference 2021*, pp. 2512–2523, 2021.

[6] G. Liang, J. Liao, W. Zhou, and J. Wen, "Integrating the pre-trained item representations with reformed self-attention network for sequential recommendation," in *2022 IEEE International Conference on Web Services (ICWS)*, pp. 27–36, IEEE, 2022.

[7] D. Cai, S. Qian, Q. Fang, J. Hu, and C. Xu, "User cold-start recommendation via inductive heterogeneous graph neural network," *ACM Transactions on Information Systems*, vol. 41, no. 3, pp. 1–27, 2023.

[8] R. Xie, Z. Qiu, J. Rao, Y. Liu, B. Zhang, and L. Lin, "Internal and contextual attention network for cold-start multi-channel matching in recommendation.," in *IJCAI*, pp. 2732–2738, 2020.

[9] N. Zhu and J. Cao, "Enhancing cross-domain recommendation through preference structure information sharing," in *2020 IEEE International Conference on Web Services (ICWS)*, pp. 524–531, IEEE, 2020.

[10] L. Wang, B. Jin, Z. Huang, H. Zhao, D. Lian, Q. Liu, and E. Chen, "Preference-adaptive meta-learning for cold-start recommendation.," in *IJCAI*, pp. 1607–1614, 2021.

[11] Z. Wang, C.-A. Sun, and M. Aiello, "Context-aware iot service recommendation: A deep collaborative filtering-based approach," in *2022 IEEE International Conference on Web Services (ICWS)*, pp. 150–159, IEEE, 2022.

[12] Q. Wan, X. He, X. Wang, J. Wu, W. Guo, and R. Tang, "Cross pairwise ranking for unbiased item recommendation," in *Proceedings of the ACM Web Conference 2022*, pp. 2370–2378, 2022.

[13] Z. Qin, S. J. Chen, D. Metzler, Y. Noh, J. Qin, and X. Wang, "Attribute-based propensity for unbiased learning in recommender systems: Algorithm and case studies," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2359–2367, 2020.

[14] G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, and K. Gai, "Deep interest network for click-through rate prediction," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 1059–1068, 2018.

[15] J. MacQueen, "Classification and analysis of multivariate observations," in *5th Berkeley Symp. Math. Statist. Probability*, pp. 281–297, 1967.

[16] X. Zhang, H. Liu, Q. Li, and X.-M. Wu, "Attributed graph clustering via adaptive graph convolution," *arXiv preprint arXiv:1906.01210*, 2019.

[17] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[18] A. Mnih and K. Kavukcuoglu, "Learning word embeddings efficiently with noise-contrastive estimation," *Advances in neural information processing systems*, vol. 26, 2013.

[19] P. O. Hoyer, "Non-negative matrix factorization with sparseness constraints.," *Journal of machine learning research*, vol. 5, no. 9, 2004.

[20] R. v. d. Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," *arXiv preprint arXiv:1706.02263*, 2017.

[21] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[22] Q. Wu, H. Zhang, X. Gao, J. Yan, and H. Zha, "Towards open-world recommendation: An inductive model-based collaborative filtering approach," in *International Conference on Machine Learning*, pp. 11329–11339, PMLR, 2021.

[23] T. Qian, Y. Liang, Q. Li, and H. Xiong, "Attribute graph neural networks for strict cold start recommendation," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[24] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[25] X. Wang, M. Bendersky, D. Metzler, and M. Najork, "Learning to rank with selection bias in personal search," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 115–124, 2016.

[26] X. Ling, W. Deng, C. Gu, H. Zhou, C. Li, and F. Sun, "Model ensemble for click prediction in bing search ads," in *Proceedings of the 26th international conference on world wide web companion*, pp. 689–698, 2017.

[27] J. Chen, H. Dong, X. Wang, F. Feng, M. Wang, and X. He, "Bias and debias in recommender system: A survey and future directions," *ACM Transactions on Information Systems*, vol. 41, no. 3, pp. 1–39, 2023.

[28] N. Kordzadeh and M. Ghasemaghaei, "Algorithmic bias: review, synthesis, and future research directions," *European Journal of Information Systems*, vol. 31, no. 3, pp. 388–409, 2022.

[29] Z. Zhao, L. Hong, L. Wei, J. Chen, A. Nath, S. Andrews, A. Kumthekar, M. Sathiamoorthy, X. Yi, and E. Chi, "Recommending what video to watch next: a multitask ranking system," in *Proceedings of the 13th ACM Conference on Recommender Systems*, pp. 43–51, 2019.