Graph Neural Networks for Default Risk Prediction and Recommendation

A thesis submitted in fulfilment of the requirements
for the degree of

Doctor of Philosophy
in
Analytics

by

Zihao Li

to

School of Computer Science
Faculty of Engineering and Information Technology
University of Technology Sydney
NSW - 2007, Australia

May 2025

Certificate of Original Authorship

I, *Zihao Li*, declare that this thesis is submitted in fulfilment of the requirements for the award of *Doctor of Philosophy*, in the *School of Computer Science, Faculty of Engineering and Information Technology* at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research was supported by an Australian Government Research Training Program (RTP) Scholarship doi.org/10.82133/C42F-K220.

Production Note:
Signature: Signature removed prior to publication.

Date: 12/05/2025

Abstract

Encouraged by the incredible success of graph neural networks on a broad spectrum of practical applications, using this technology for credit risk prediction and recommendation has become ubiquitous recently. Specifically, home credit default risk prediction aims to detect clients and loan applications that have the potential risk of failing to repay a loan or meet contractual obligations. It is essential for banks and financial institutions to keep good health in management and operations. Recommendation systems endeavor to assist users in finding valuable information or potential products they might prefer from the massive amount of items, rendering the decision-making process easier. These two tasks encounter a huge challenge of graph data processing and mining, and both of them attract widespread attention from academia and industry. In this study, we mainly focus on the graph neural networks for default risk prediction and recommendation. More concretely, we apply graph neural networks to alleviate the data missing issue via information propagation and aggregation of similar records. Besides, multi-view graphs are designed for small data augmentation to tackle the unbalanced and skewed distribution problem. So as to the session-based recommendation, we first give a comprehensive review of the graph and sequential neural networks in this task. Then, a dual graph neural network is proposed to capture the implicit and explicit relationships among the external and internal session connections, which will be fused together for the final recommendation. Also, considering the skewed item distribution in the recommendation system, a reweighing and reembedding strategy is applied to alleviate the tail item problem in text-based or image-based recommendation. We conducted extensive experiments with regard to the default risk prediction and recommendation tasks on three and six datasets against fifteen and twenty-four baselines, respectively, to verify the effectiveness of our methods. We believe the proposed multi-view and dual graph neural networks are capable of transferring and facilitating a wide range of real-world domains, including medicine, traffic, social networks, and beyond.

Keywords: Graph Neural Networks, Sequential and Session-based Recommendation, Default Risk Prediction

AUTHOR'S DECLARATION

I, *Zihao Li* declare that this thesis, submitted in partial fulfilment of the requirements for the award of Doctor of Philosophy, in the *University of Technology Sydney*, *Faculty of Engineering and Information Technology* at the University of Technology Sydney, Australia, is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis. This document has not been submitted for qualifications at any other academic institution. This research was supported by an Australian Government Research Training Program (RTP) Scholarship doi.org/10.82133/C42F-K220.

SIGNATURE: Production Note: Signature removed prior to publication.

DATE: 7th May, 2025

PLACE: Sydney, Australia

ACKNOWLEDGMENTS

I would like to express my heartfelt gratitude to everyone who has inspired, encouraged, and supported me throughout the journey of completing this thesis.

First and foremost, I extend my sincerest thanks to my supervisor, Dr. Xianzhi Wang, and Prof. Guandong Xu. Their guidance and insightful suggestions have been invaluable not only for my Ph.D. journey but also for my future aspirations. I first met Professor Guandong Xu during my first year of postgraduate studies at the University of Chinese Academy of Sciences while taking his Recommender Systems class. His friendliness and warm heart give me good vibes with regard to my academic pursuit. I am sincerely grateful to him for providing me the opportunity to pursue my PhD at UST with such honor. I also want to extend my thanks to Dr. Xianzhi for his invaluable support and guidance. I still vividly recall the first email I received from him, in which he discussed my future study plans at UTS, expecting me to become more competitive as I approach graduation. I am deeply moved by the sincerity and kindness I have encountered. He respects my ideas, opinions, and choices, always tolerating my mistakes and standing by my side with unwavering support and constructive suggestions when I need them most. Through my experiences with Dr. Xianzhi, I have come to truly appreciate the equality and care that exists in the relationship between supervisor and student. It has been a privilege to be a student of Dr. Xianzhi and I regret that three years of study is too short to let me learn more from him.

I am also grateful to the University of Technology Sydney for its technical and logistical support, which facilitated my research endeavors.

I would like to thank my colleagues, Dr. Chao Yang, Miss Yakun Chen, and Miss Yicong Li, for their unwavering assistance during challenging times in both my personal and academic life. I truly cannot imagine navigating these challenges without their support. Additionally, I would like to acknowledge team members and friends in the DSMI group, whose suggestions, comments, and discussions in the regular group meetings inspired me a lot.

Last but not least, I extend my appreciation to my family and friends for their constant encouragement and belief in me.

Thank you all for being a part of this journey.

List of Publications

Related to the Thesis:

1. **Zihao Li**, Xianzhi Wang, Lina Yao, Yakun Chen, Guandong Xu, and Ee-Peng Lim. "Graph neural network with self-attention and multi-task learning for credit default risk prediction." In International Conference on Web Information Systems Engineering (WISE 2022), pp. 616-629. 2022.

2. **Zihao Li**, Xianzhi Wang, Chao Yang, Lina Yao, Julian McAuley, and Guandong Xu. "Exploiting explicit and implicit item relationships for session-based recommendation." In Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM 2023), pp. 553-561. 2023.

3. **Zihao Li**, Yakun Chen, Xianzhi Wang, Lina Yao, and Guandong Xu. "Multi-view GCN for loan default risk prediction." Neural Computing and Applications (2024): 1-14.

4. **Zihao Li**, Chao Yang, Yakun Chen, Xianzhi Wang, Hongxu Chen, Guandong Xu, Lina Yao, and Michael Sheng. "Graph and sequential neural networks in session-based recommendation: A survey." ACM Computing Surveys 57, no. 2 (2024): 1-37.

5. **Zihao Li**, Yakun Chen, Tong Zhang, Xianzhi Wang. "Reembedding and Reweighting are Needed for Tail Item Sequential Recommendation.", Web Conference (WWW 2025), pp. 4925-4936. 2025.

Others:

6. Yakun Chen, **Zihao Li**, Chao Yang, Xianzhi Wang, Guodong Long, and Guandong Xu. "Adaptive graph recurrent network for multivariate time series imputation." In International Conference on Neural Information Processing (ICNIP), pp. 64-73. 2022.

7. Chao Yang, Yakun Chen, **Zihao Li**, and Xianzhi Wang. "Exploring the Effectiveness of Positional Embedding on Transformer-Based Architectures for Multivariate Time Series Classification." In International Conference on Advanced Data Mining and Applications (ADMA), pp. 34-47. 2023.

8. Yakun Chen, Ruotong Hu, **Zihao Li**, Chao Yang, Xianzhi Wang, Guodong Long, and Guandong Xu. "Exploring explicit and implicit graph learning for multivariate time series imputation." Engineering Applications of Artificial Intelligence, 127 (2024): 107217.

9. Chao Yang, Yakun Chen, **Zihao Li**, Xianzhi Wang, Kaize Shi, Lina Yao, Guandong Xu, and Zhongwen Guo. "Deep Multimodal Learning for Time Series Analysis in Social Computing: a Survey". International Journal of Multimedia Information Retrieval (IJMIR), 14 (2025): 15.

10. Yakun Chen, **Zihao Li**, Chao Yang, Xianzhi Wang, and Guandong Xu. "Large Language Models are Few-shot Multivariate Time Series Classifiers." Data Mining and Knowledge Discovery, 39.5 (2025): 66.

11. Tong Zhang, Nitin Bisht, **Zihao Li**, Guandong Xu, and Xianzhi Wang. "SarRec: Statistically-guaranteed Augmented Retrieval for Recommendation." In International Conference on Information and Knowledge Management (CIKM 2025) (Accepted).

12. Tong Zhang, Nitin Bisht, **Zihao Li**, Xianzhi Wang, Xiuwen Gong, and Guandong Xu. "Adaptive Retrieval-Augmented Generation for LLM-based Sequential Recommendation." Submitted to The 40th Annual AAAI Conference on Artificial Intelligence (AAAI 2026).

13. Chao Yang, Yakun Chen, **Zihao Li**, Xianzhi Wang, and Zhongwen Guo. "Positional Embedding May Not Be All You Need: An Empirical Study on Positional Embedding for Transformer". Submitted to Expert Systems with Applications (ESWA).

TABLE OF CONTENTS

[1]This Chapter is based on our published work: Graph neural network with self-attention and multi-task learning for credit default risk prediction.

[2]This Chapter is based on our published work: Multi-view GCN for loan default risk prediction.
[3]This Chapter is based on our published work: Graph and sequential neural networks in session-based recommendation: A survey.

[4]This Chapter is based on our published work: Exploiting explicit and implicit item relationships for session-based recommendation.

[5]This Chapter is based on our published work: Reembedding and Reweighting are Needed for Tail Item Sequential Recommendation.

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

1.1 Background

Graph data is ubiquitous in our daily lives and industry applications, including social networks, biological systems, recommendation systems, financial risk prediction, and traffic management. Distinct from general tabular data structures, graph data provides a powerful and flexible framework for representing, mining complex and dynamic relationships among entities. In our research, we specialize in graph neural networks for credit risk prediction and recommendation, the two hot-spot research topics that attract widespread attention from both academics and industry. We dive into tailored graph neural network structures in these two areas, aiming to alleviate the problem of unbalanced data distribution.

Financial credit risk represents the potential that a borrower may default or be unable to repay lenders. Aiming to assess the risk of borrower defaults, loan default risk prediction, or credit risk prediction [101] is crucial for financial institutions. In general, credit risk assessment can be recognized as the following: given a set of loan applications with personalized information and various financial indicators over a specific period, predicting the likelihood that a client will fall into a high-risk category within the coming years. This problem is typically divided into two categories: credit rating (or credit scoring) and loan default prediction. Both tasks are approached similarly, often as binary classification challenges, i.e., classified as either high/low credit risk [25]. In today's financial system, the sheer volume of loan applications overwhelms human

reviewers, leading to increased wait times. To streamline this process, machine learning techniques [74, 94, 129] are increasingly utilized to predict loan defaults based on borrower profiles, which include factors like occupation, income, and credit history. Explainable models, such as Logistic Regression [45] and Gradient Boosted Decision Trees (GBDT) [62], are preferred due to their transparency – a critical requirement in light of growing regulations on financial algorithms. Attributed to the powerful pattern extraction and feature representation ability, deep neural networks [82, 116] have become prevalent and emerging in this area.

Recommendation system [189] is a prominent solution to alleviate the problem of information overload [57]. Given the rapid advancement of Intent, this technology spans a wide range of application scenarios, including E-commerce, social media, and entertainment. Conventional recommendation solutions focus on statistical methods via collaborative filtering, including item-based [193], user-based [228], and the hybrid [145]. Specifically, the collaborative filtering methods endeavor to find similar users or items based on the user-item interaction records, and then recommend the items that the user might prefer from similar users. Apart from that, machine learning attempts, including KNN [115], Bayes rules [187], and matrix factorization [110] are also adopted for recommendation at the early stage. Motivated by the remarkable results of deep neural networks in computer vision [112], feed-forward neural networks [192], conventional neural networks [218], sequential neural networks [99], graph neural networks [78], Transformers, and their variants [34, 119, 134, 202] are erupting successively.

Overall, both credit application records, as well as user-item interaction history, can be structured as complicated graph data. In this thesis, we mainly focus on graph neural networks for credit risk prediction and recommendation. However, reviewing existing solutions, they neglect the dynamic and implicit relationships within graphs. Moreover, all these studies suffer from the issues posed by skewed data distributions, which are significant yet insufficiently explored in the existing works.

1.2 Thesis Overview

In a practical scenario, the raw application records for loan default detection contain a considerable amount of missing data and face a significantly unbalanced distribution (i.e., default samples represent only a small fraction of the total applications), resulting in suboptimal model performance. To overcome these two problems, we first construct a loan application graph, thus, the external knowledge from similar clients can be in-

troduced as auxiliary information via graph neural networks for default risk prediction. We further propose multi-viewed loan application graphs, which can adjust the graph structures dynamically based on the similarity threshold. Consequently, we could attain a group of representations derived from the multi-view graphs for small sample representation augmentation. As for recommendation systems, existing graph neural network-based methods only model the explicit connections between items, whereas the implicit connections are also important in session-based and sequential recommendations. Regarding these two connections, we propose dual-graph neural networks to capture the internal and external relationships, respectively, and both of them will be utilized for recommendation. In addition, owing to the popularity bias, the skewed distribution in recommendation systems is caused by the tail items. Instead of the data augmentation strategy in default risk prediction, we propose another solution in sequential recommendations, i.e., reweighting and reembedding. Therefore, this solution allows the model to pay more attention to the tail item in the training process. The detailed contents of this thesis with regard to the graph neural networks in default risk prediction and recommendation are as follows:

- So as to the default risk prediction, we first review the existing methods, which can be split into three categories: statistics-based methods, machine learning-based methods, and deep learning-based methods. We will further conduct a detailed investigation with regard to graph neural networks for default risk prediction, as this research line is most related to our research.

- In addition, we observe that a great number of client application information is incomplete, the missed historical records will hinder the experts or models from giving a precise assessment of the applications. To alleviate the problem of missing data, we consider that clients possessing similar personal information can provide some auxiliary information for the target applicant risk prediction. Consequently, based on the similarity of application records, we organize them as a graph to exploit the potential connections. Then, a graph neural network is adopted. Attributed to the information propagation and aggregation, the missing data representation can be augmented effectively for accurate prediction.

- Besides the data missing problem, the distribution of applications is extremely unbalanced and skewed in the real world, i.e., most of the records are health and non-default risks, while very few samples are identified as default applications. The

unbalanced data distribution also inherently induces the positive samples to dominate the parameter update directions while ignoring the small sample optimization, which is critically important for risk prediction. To alleviate this issue, on the fundamentals of the first research work, we further propose multi-view augmented graph networks. Specifically, we set a series of similarity thresholds to dynamically control and adjust the graph structures. Thus, multiple representations can be derived from the multi-view graph structures, and they will be used for small data augmentation for skewed distribution rebalanced. Extensive experiments on three public datasets against fifteen baselines illustrate the effectiveness of our proposed solutions.

- As for graph neural networks in recommendation systems, we first give a comprehensive overview of the recent works on session-based recommendation, which includes five main contents: (1) we standardize the concepts and definitions with regard to the session-based recommendation and introduce various graph structures that are commonly used in this task; (2) we summarize the features of session-based recommendation and compare the similarities and distinctions against sequential recommendation; (3) we propose unified frameworks of sequential neural networks and graph neural networks for SR, respectively, thus, a systematic category is proposed to organize the existing works; (4) a comprehensive analysis and comparisons of the properties of these two mainstream methods are presented, the overall pipeline and key modules in these methods are also introduced in detail; (5) the popular public datasets and commonly used evaluation metrics are introduced, therefore, detailed performance and computation complexity comparison are conducted.

- Note that most existing GNN-based session-based recommendation solutions focus solely on the explicit connections between two items (i.e., neighbors on the timeline) while ignoring the implicit connections among all the items, which are equally important for recommendation. To capture these two relationships, we propose external and internal graph neural networks. Consequently, local and global information from items can be fused for representation enhancement and recommendation.

- Moreover, the skewed distribution problem, also known as the tail item problem (i.e., a small group of items receives the most exposure, while a large number of items are unpopulated and have no or very few interactions from users), is a

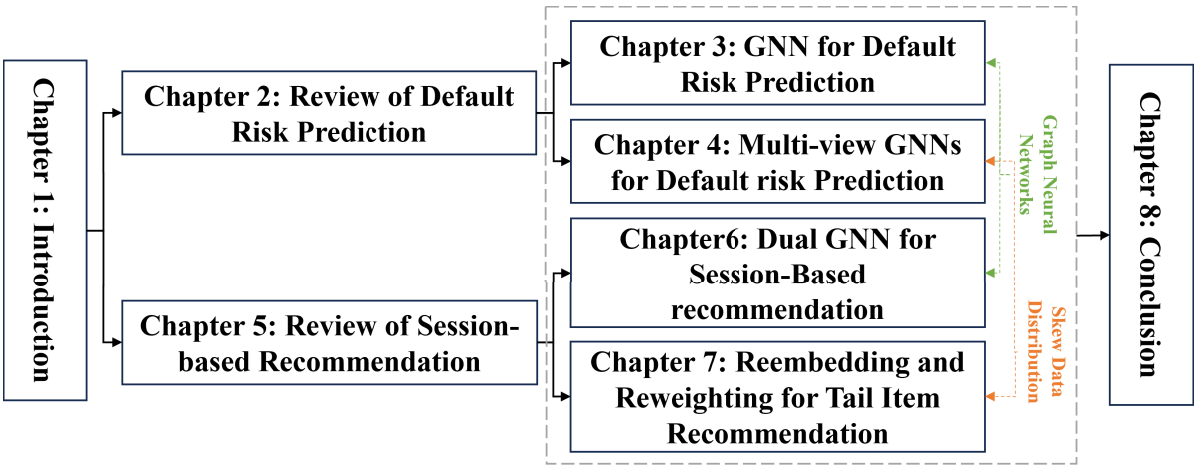


Figure 1.1: The diagram of chapter organization.

chronic and prominent problem in sequential recommendation. Different from the solution in default risk prediction that leverages data augmentation to rebalance the data distribution, we apply another attempt, i.e., reembedding and reweighting, to alleviate this issue. This solution guides the model to focus more on the tail item when model training by reweighting the losses and adjusting the loss distribution adaptively, thus, the accuracy of the long-tail item prediction can be improved. We conducted extensive experiments on six real-world datasets against twenty-four baselines to verify the effectiveness of our proposed method on recommendation tasks. Embedding visualizations further illustrate the merits of our method in the uniformity and alignment of these two properties.

Figure 1.1 presents the overall structure of the subsequent chapters. The remaining chapters can be categorized into two groups according to the downstream tasks: default risk prediction (Chapters 2, 3, and 4) and recommendation (Chapters 5, Chapter 6, and 7). Across these chapters, graph neural networks serve as the primary methodological framework. Besides, Chapters 4 and 6 address the challenge of skewed data distributions inherent in both tasks. To sum up, in Chapter 2 we overview the existing methods for loan default risk prediction. Chapter 3 and Chapter 4 introduce the graph neural networks and multi-view graph neural networks to relieve the missing values and skewed data distribution problems in loan default risk prediction, respectively. In Chapter 5, we summarize the graph and sequential neural networks in session-based recommendation. Chapter 6 introduces our proposed dual graph neural networks for implicit and explicit connections modeling and session-based recommendation. Chapter 7 further introduces

our reembedding and reweighting strategy to alleviate the tail item prediction problem in sequential recommendation. Finally, in Chapter 8, we summarize this thesis along with several future directions.

1.3 Dataset Used

We select three and eight public datasets, respectively, for the evaluation of default risk prediction and recommendation. All of them are widely used in these two tasks. Specifically, for default prediction, we select *Home Credit Default Risk dataset*, *Lending Club*, and *PDD*

- **Home Credit Default Risk**[1] aims to provide loan services to individuals with little or no credit history. To accurately assess the client's creditworthiness and capability of fulfillment, financial institutions tend to refer diverse auxiliary information, collecting complex, multi-source, and heterogeneous application records. Therefore, the dataset includes applicant information such as demographics, income and expenditure, credit history, loan records, and repayment behavior.

- **Lending Club**[2] is a large-scale P2P lending dataset containing millions of loan records with 151 attributes spanning 2007,Äì2018.

- **PPD**[3] is a public loan default prediction dataset released from a Heywhale community competition. Each record consists of numerous continuous and categorical attributes, covering borrower demographics, login activity, and modification history.

As for the recommendation, we select *Diginetica*, *Yoochoose*, *Gowalla*, *Last.FM*, *Amazon Beauty*, *Amazon Toys*, and *Amazon Sports*.

- *Diginetica* is a personalized e-commerce dataset from the CIKM CUP 2016, containing transaction histories suitable for session-based recommendation. Following [19, 30], we used the sessions from the last week for testing.

- *Yoochoose* is an e-commerce clickstream dataset from the RecSys Challenge 2015, covering six months of user interactions. We followed [19, 30] to split the data. Due

[1] `https://www.kaggle.com/c/home-credit-default-risk/overview`
[2] `https://www.kaggle.com/wordsforthewise/lending-club`
[3] `https://www.heywhale.com/home/competition/56cd5f02b89b5bd026cb39c9/content/1`

to its large scale, we adopted two subsampled training sets, using the most recent 1/64 and 1/4 of all sessions, denoted as "Yoochoose1/64" and "Yoochoose1/4".

- *Gowalla* is a point-of-interest recommendation dataset. Following [19, 69], we retained the top 30,000 locations and segmented user check-ins into sessions by splitting gaps longer than one day.

- *Last.FM* is a music artist recommendation dataset. Following [19, 69], we kept the top 40,000 artists and grouped user interactions into sessions of 8 hours.

- *Amazon Beauty, Toys, Sports*. We select three subcategories, i.e., *Beauty*, *Toys*, and *Sports*, from Amazon datasets[4] [161] for performance evaluation, as they contain enriched item information (e.g., item title, description, and image, and so on) and user information (e.g., user ratings and reviews, and so on).

1.4 Key Contributions

The main contributions of this thesis include: (i) novel graph neural network architectures for credit risk prediction and recommendation; (ii) two effective solutions to address the skewed data distribution problem; and (iii) a comprehensive overview of graph and sequential neural networks for session-based recommendation. In a nutshell, we make the following contributions in this thesis:

- We adopt the graph neural network and the multi-view graph neural network to address the missing value and unbalanced data distribution problems in loan default risk prediction. Experimental results on three datasets against state-of-the-art verify the effectiveness of our proposed methods.

- We provide a comprehensive review of over 150 papers focusing on graph and sequential neural networks for session-based recommendation. We further compare the performance and computational complexity of the representative modules in session-based recommendation.

- We propose internal and external graph neural networks for session-based recommendation. We conduct extensive experiments on four public datasets against eleven baselines, demonstrating the superiority of our method.

[4] `https://nijianmo.github.io/amazon/index.html`

- We propose an efficient yet effective plugging, i.e., reembedding and resampling, to alleviate the tail item recommendation problem. This strategy enables efficient application to other base models for performance improvement.

CHAPTER 2

REVIEW OF DEFAULT RISK PREDICTION

Credit default risk refers to the possibility of a loss resulting from a borrower's failure to repay a loan or meet contractual obligations. It is a major concern for any bank and financial institution in making loan decisions [1]. Bad loans can cause banks problems with their capital adequacy and, at worst, lead to default. Bad loans also risk impairing long-term economic growth and leading to greater uncertainty and instability in the banking and financial systems. Therefore, it is highly necessary to assess borrowers' repayment abilities before authorizing a loan, which calls for accurate credit default risk prediction. In this Chapter, we summarize the motivated works proposed for loan default risk prediction, including machine learning, deep learning, and graph-based methods.

2.1 Machine Learning Methods

In the early stage, tremendous efforts have been devoted to developing qualitative and quantitative loan default risk assessment for loan default prediction [302]. Based on the credit scoring systems and the rule of thumb, the credit score of borrowers could be obtained for loan application authorization [47, 75, 173, 210]. For example, the widely adopted '5C principle' [1] requires domain professionals to evaluate borrowers' default risk by manually evaluating borrowers on five aspects: *character*, *capital*, *capacity*, *collateral*, and *conditions*. Serrano-Cinca et al. [196] pointed out that the loan purpose, annual income, and current housing situation remain strongly correlated with credit score assessment. However, it was challenging to acquire precise loan

default risk evaluation results due to the various risks and the complexities of dependencies between numerous influencing factors. Hence, the machine learning methods, e.g., tree-based classifiers [4, 204], support vector machines (SVM) [8, 94, 96, 105, 244], neural networks (NN) [96, 117, 184], ensemble-based methods [8, 155], hybrid approaches [39, 94, 117, 258], and others [7, 162, 180], have been adopted for loan default risk prediction. Specifically, building on the principle of "divide and conquer", the decision tree algorithm systematically selected various attributes and integrates them to enhance the purity of distinct classes and improve classification accuracy [181]. Owing to the transparency of the training mechanism, remarkable explainability, and effectiveness in classifying structural data, the decision tree was a popular solution in credit default risk prediction [4, 44, 204]. Considering the capability limitation of the feature representation of a single tree, ensemble approaches, i.e., bagging and boosting, were leveraged for credit default risk prediction. Depending on whether the type of individual learner is the same, the integration method can be categorized into homogeneous learning and heterogeneous learning, also known as boosting and bagging. Bagging methods (e.g., Random Forest [15]) trained multiple classifiers simultaneously and then combined them to make the final prediction [194]. Given the inherent dependencies among individual models, bagging methods facilitated distributed parallel training and computational efficiency improvement. For instance, as a typical model of bagging strategy, Random Forest selected samples and attributes randomly to construct and integrate decision trees. Uddin et al. [222] leveraged Random Forest in micro-enterprises credit default risk modeling. Zhu et al. [313] concluded that Random Forest has better accuracy than other machine learning methods like logistic regression, decision trees, and SVM. In contrast, boosting methods (e.g., XGBoost [28], LightGBM [104]) applied individual models in a chain, where each model takes as input the result of the previous model [121, 129]. The cascade training strategy achieved a better performance against bagging attempts in general Aleksandrova et al. [4] evaluated several popular machine learning algorithms for P2P credit scoring and argue that ensemble classifiers (e.g., XGBoost, GBM, and Random Forest) outperform non-ensemble models (e.g., logistic regression, decision tree, and multilayer perception). Yeh and Lien highlighted that neural networks achieve the best result on default risk prediction, in contrast to five other machine learning methods, including K-nearest neighbors, logistic regression, classification trees, discriminant analysis, and naive Bayes [278]. However, Huang et al. concluded that SVM is superior to neural networks and other models [94]. Coser et al. [44] pointed out that logistic regression and Random Forest obtained the best performance among various

machine learning methods for default risk prediction. To exploit the advantages of each model thoroughly for performance and robustness improvement, a stacking strategy was proposed. Li et al. [121] designed a stacking framework that integrated XGBoost, SVM, and Random Forest for P2P default risk prediction and experimentally showed that the model fusion algorithm has better adaptability and accuracy.

Apart from ensemble-based attempts, Wang et al. proposed a fuzzy support vector machine that enhanced generalization capabilities by treating each sample as belonging to both positive and negative classes, albeit with varying probabilities. This approach not only improved the model's flexibility but also maintained its robustness against outliers [244]. Yang [274] presented an incremental kernel learning method for credit scoring. With this approach, the scoring model can be adjusted via an online update procedure. Huang et al. [95] proposed a two-stage genetic programming (2SGP) method, which can incorporate the IF-THEN rules with neural network training for credit score prediction. Lee et al. [117] integrated neural networks with a traditional analysis approach and proposed a two-stage hybrid model. First, LDA was applied for predictor feature selection. Then, the selected features and predicted scores were further fed into a neural network to improve the accuracy of the prediction results. Zhao et al. first applied the GBDT model, which combines both static features and dynamic features for credit assessment. Then, two objective functions, i.e., weighted objective and multi-objective, were proposed to select portfolios for lenders. Furthermore, the author also designed two algorithms, namely DPA and EVA, for these two objectives optimization [303].

2.1.1 Deep Learning Methods

Although conventional machine learning methods achieved promising performance for credit default risk prediction, these models heavily rely on the quality of feature engineering, i.e., the results for the same dataset can vary enormously even with a slight change of selected features [72]. In addition, it is non-trivial for such models to handle complicated data due to the limited feature representation capacity. To resolve these issues, deep learning methods were developed recently. Babaev et al. applied RNN for the credit loan application task [5]. Wang et al. [243] used LSTM for P2P lending risk prediction. Kvammea et al. applied a convolutional neural network for mortgage default prediction [113]. Tan et al. developed a deep learning approach for charge-off and prepayment risk prediction on P2P lending dataset [216]. To be specific, the authors first generated the risk grades based on loan status, survival time, and loan term. Then, the deep learning model transformed the hierarchical grades prediction into multiple binary

classification subtasks for risk prediction. Due to the scarcity of historical lending data, accurately predicting outcomes presents a significant challenge. Therefore, Suryanto et al. [215] employed transfer learning techniques to mitigate the limitations imposed by insufficient historical data. Considering the interpretability of deep learning methods, Liu et al. proposed a novel automatic feature crossing method called DNN2LR, which applied a deep neural network for feature extraction. Then, the extracted features were further fed into a Logistic Regression model for loan credit risk assessment [147]. However, existing deep learning models rarely comprehensively use full-spectrum, multi-source, heterogeneous data for risk assessment. None of them establishes an effective approach to learning effective representations of loan applications or explicit connections among the applications. They commonly face challenges in dealing with incomplete profiles of applicants and missing information in loan applications. In this regard, we applied self-attention with multi-task learning for application record representation and also considered graph neural networks to capture the connection among similar applications and alleviate the missing values problem.

Maria et al. [164] built connections between borrowers based on the geographic locations or economic activities for borrowers' connections construction, then a multi-layer personalized PageRank model was applied for credit default risk prediction.

2.1.2 Graph-based Methods

As the loan application records encompass implicit connections naturally, graph-based methods were devoted to loan default prediction. Experiments further demonstrated the effectiveness of graph-based features [48]. For instance, Cui et al. converted the features into a graph-based representation to capture global topological information. Subsequently, random walk algorithms and C-SVM were employed for graph-based feature representation and credit risk prediction, respectively [48]. Zhong et al. proposed an attributed heterogeneous information network for credit risk prediction. First, user behaviors were adopted for node representation learning via the meta-path method. Then, the attention mechanism was applied for node classification [306]. Guo et al. pointed out that existing works mainly focused on feature interactions while sample relations were ignored. Hence, the authors constructed a multiplex graph and applied a graph neural network to learn an enhanced representation for each sample [71]. Considering the complexity of financial scenarios, Hu et al. developed an Attributed Multiplex Graph (AMG) to model various relations and the rich attributes of nodes and edges simultaneously. Through local structure and multiplex relations modeling,

Table 2.1: Comparative of mainstream technologies.

Category	Strengths	Limitations	Suitability
Credit Scoring Systems	High interpretable ability; efficient; expert knowledge incoporation	Labour-intensive; time-consuming; poor transferability	High reliability and interpretability scenario
Machine Learning (ML)	Interperatable, efficient with small-to-moderate datasets	Poor at relational modeling; limited capacity for complex data	Tabular financial features
Deep Learning (DL)	High representational capacity; models nonlinear/temporal dependencies	Requires large datasets; black-box; limited relational modeling	Big data; Sequential or high-dimensional data
Graph-Based Methods (GNNs)	Relational and structural dependencies modeling; robust to sparsity	Higher computational cost; graph construction required	Imbalanced and relational datasets

the AMG achieved the state-of-the-art for users' credit risk prediction on a large-scale real-world dataset [89].

In general, traditional machine learning methods mostly rely on the statistical features extracted from various aspects. However, such complicated information is hard to capture by these feature-based approaches in real-world scenarios. Although there have been a few research endeavors to explore an end-to-end method for credit default risk prediction via deep learning methods very recently, these models do not conduct experiments on open, large-scale datasets. Although the existing works propose graph neural networks for loan default prediction, they mainly rely on users' social relations for graph construction. In the real scenario, the social relationships between users may be unknown due to privacy protection limitations. Thus, it is intractable to construct a graph based on existing methods. In addition, the above methods model the heterogeneous attributes of nodes and edges with a single graph, which is insufficient to represent such complicated information. Moreover, the imbalanced data distribution prevents the model from achieving optimal results. Consequently, in our work, we considered both the heterogeneity of attributes and graph structures and constructed multi-graphs for small sample augmentation and loan application.

To sum up, the advantages, limitations, and suitability of conventional credit scoring systems, machine learning, deep learning, and graph neural network methods for default risk prediction are summarized in Table 2.1. We discuss these types of solutions from different perspectives: explainability, efficiency, capability of complicated data structure modeling.

CHAPTER 3

GNN FOR CREDIT DEFAULT RISK PREDICTION

3.1 Introduction[1]

Traditional risk assessment highly relies on experts with professional knowledge and relevant experience to assess loan requests against specific business models and rules. Nevertheless, it helps to incorporate both qualitative and quantitative measures, which makes credit default risk assessment even more complicated, labor-intensive, time-consuming, and prone to personal bias. The availability of big financial data related to personal and home credit offers the opportunity for automating credit default risk assessment with predictive models. Until recently, the related research has been focusing on traditional machine learning techniques, e.g., Support Vector Machine (SVM) [94], decision tree [204], Random Forest (RF) [155], and XGBoost [121]. These models' performance highly depends on the quality of feature engineering, which requires high domain expertise to incorporate diverse sources and forms of data. Recently, deep learning has shown great potential in addressing the above challenges, thanks to its capability to capture complex, non-linear relations from massive data [208, 209]. It has proven successful in various domains, such as computer vision natural language processing, speech recognition, and recommendation systems [49, 116, 280].

However, there have been limited studies on credit default risk prediction based on deep learning techniques. Besides, existing deep learning methods need to address

[1]This Chapter is based on our published work: Graph neural network with self-attention and multi-task learning for credit default risk prediction.

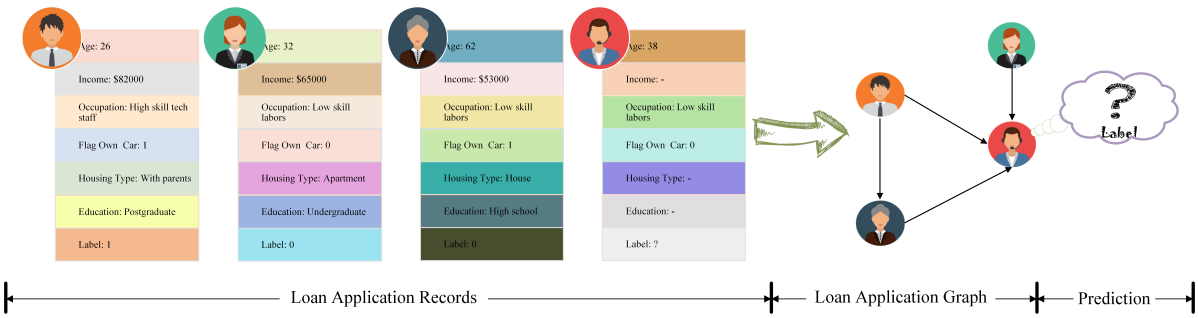


Figure 3.1: A toy example. Loan application records are represented as nodes in a loan application graph. Then, credit default risk prediction can be transformed into a node classification problem.

several challenges for accurate performance. First, it requires incorporating all sources of clues about users' backgrounds, credit histories, investments, etc., to enhance predictions. Second, certain users may have insufficient or incomplete information (e.g., non-existence of credit histories) for the prediction task, making it necessary to leverage the information about other users who have more useful information to help improve the prediction results. The current studies cannot effectively leverage multiple aspects of clues (e.g., heterogeneous information in applicants' profiles, relations among loan applications) to overcome the challenges posed by incomplete profiles and missing values, which greatly impair the prediction accuracy [164].

We argue that a loan application graph can be designed based on raw application records for prediction. Intuitively, similar application records are likely to incur similar risk levels; so the similar neighbors can be considered as auxiliary information to alleviate the issue of missing values for credit default risk prediction. As shown in Figure 3.1, we could construct a loan application graph based on the similarity between each client's historical records first. Hence, the neighbor's information can be introduced as external information to alleviate the missing values problem. Moreover, our proposed graph neural network-based methods are capable of offering explanatory insights into the prediction results, owing to the inherent ability of neighboring nodes' information aggregation. This property allows us to leverage similar records as auxiliary information, thereby enhancing prediction accuracy. In addition, interpretability can be further achieved by analyzing the connection strengths among neighbors, where stronger connections indicate a more significant influence on the current node.

Overall, the credit default risk of each record can be predicted via node classification method. We, thereby, propose a novel Graph neural network with input attributes Self-

attention and Multi-task learning (SaM-GNN), which comprehensively incorporates self-attention, graph neural networks, and multitask learning for accurate credit default risk prediction. In a nutshell, we make the following contributions:

- We construct an undirected graph for loan applications and combine self-attention and graph convolution networks for representation learning in our model. The self-attention mechanism enables effective feature representation, and graph convolution networks allow for aggregating similar information via a graph structure to improve the model's robustness to missing values in the input.
- We design two parallel tasks for multi-task learning based on shared feature representation: a decoder module for input reconstruction and a classification module for credit default risk prediction. The two tasks are jointly trained to optimize feature representation and prediction results simultaneously.
- We conducted experiments on two real-world credit default risk prediction datasets. Our experimental results show a significant performance improvement of our approach over state-of-the-art methods. Besides, the feature representations of loan applications output by our approach help improve the performance of existing ensemble methods.
- We will make our data and code public, including detailed parameter configurations for all the methods, to ensure reproducibility.

3.2 Proposed Method

Our proposed model (shown in Fig. 3.2) works as follows: It first generates the embedding of categorical data (e.g., gender, suite type, education) and applies the self-attention mechanism to the embedding and numeric data (e.g., income total and goods price) for feature representation. Then, the resulting representations are concatenated and updated via a graph convolution constructed based on the similarity between loan applications to alleviate the impact of missing values. Finally, a decoder module and a classification module (consisting of multiple fully connected layers) are jointly trained to generate feature representation and simultaneously predict credit default risk. To sum up, our proposed framework mainly includes four modules that correspond to the above three steps, respectively: 1) **Self-attention Module** for feature representation generation, 2) **Graph Convolution Module** for graph construction and neighbor representation

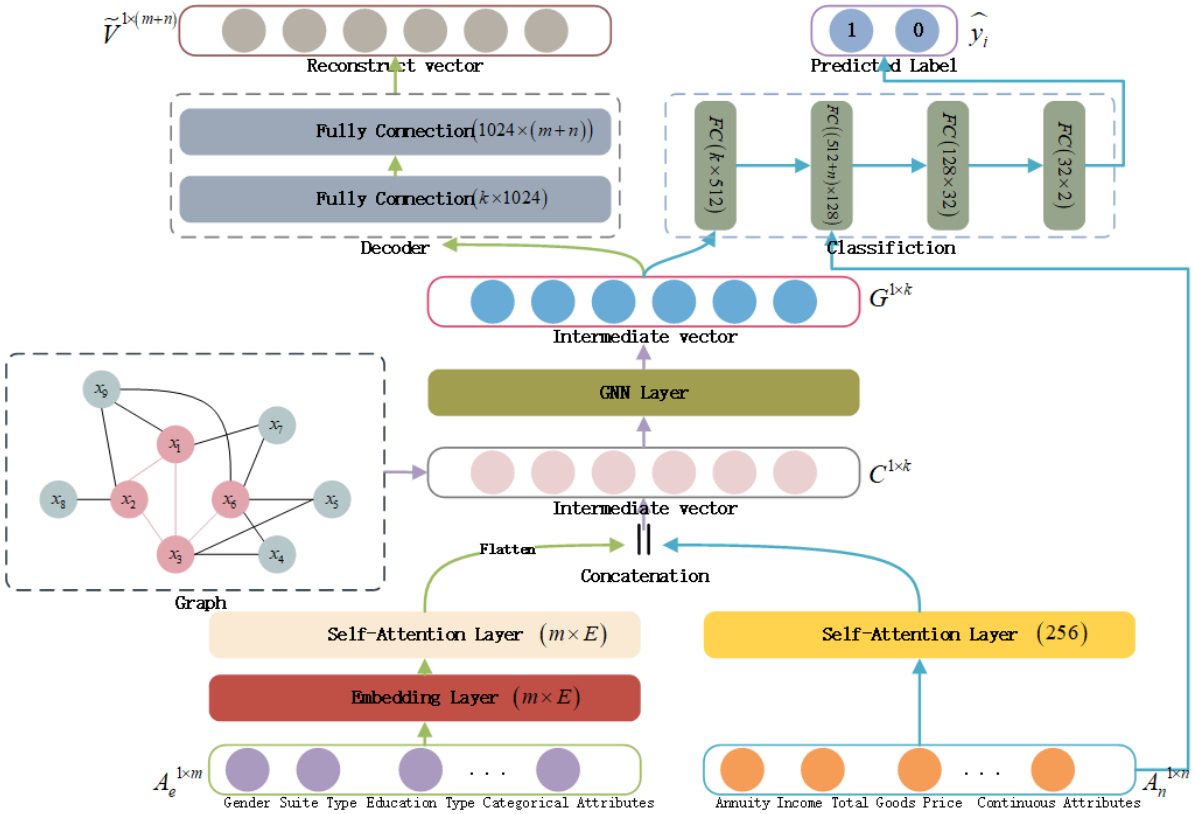


Figure 3.2: The framework of SaM-GNN. $A_e^{1\times m}$. $A_n^{1\times n}$ are categorical attributes and numeric attributes, respectively, where m, n are the corresponding numbers. E is the embedding size, and $k = m \times E + 256$. C and G are intermediate vectors before and after similar information aggregation via the graph convolution block. $\tilde{V}$ is the reconstructed input, and $\hat{y}$ is the predicted label.

aggregation update, 3) Decoder module, and 4) Classification module for intermediate vector reconstruction and credit default risk prediction.

3.2.1 Self-Attention Module

Benefit to the powerful feature representation ability, the self-attention mechanism has become a standard component in many deep neural network architectures. Here, we use self-attention to generate feature representations for categorical and numeric inputs. Let $X_e = \{x_{e_0}, x_{e_1}, ..., x_{e_m}\}$ be categorical attributes and $X_n = \{x_{e_0}, x_{e_1}, ..., x_{e_n}\}$ be numeric attributes in the input. Suppose $\mathbf{E_i} \in \mathbb{R}^{1\times d}$ is the embedding of categorical attribute x_{e_i}, where d is the embedding size. The feature representations of categorical attributes are

as follows:

$$
\begin{cases}
\mathbf{Q_e} & = \mathbf{E_e}\mathbf{W_q} + \mathbf{b_q} \\
\mathbf{K_e} & = \mathbf{E_e}\mathbf{W_k} + \mathbf{b_k} \\
\mathbf{V_e} & = \mathbf{E_e}\mathbf{W_v} + \mathbf{b_v} \\
\text{Attention}(\mathbf{Q_e}, \mathbf{K_e}, \mathbf{V_e}) & = \text{softmax}(\frac{\mathbf{Q_e}\mathbf{K_e^T}}{\sqrt{d_k}})\mathbf{V_e}
\end{cases} \tag{3.1}
$$

where $\mathbf{E_e} \in \mathbb{R}^{m \times d}$ is the embedding of categorical attributes. $\mathbf{W_q}, \mathbf{W_k}, \mathbf{W_v} \in \mathbb{R}^{d \times d}$ and $\mathbf{b_q}, \mathbf{b_k}, \mathbf{b_v} \in \mathbb{R}^{1 \times d}$ are learning weight matrices. $1/\sqrt{d_k}$ is the scaling factor. Similarly, the feature representations of numeric attributes are as follows:

$$
\begin{cases}
\mathbf{Q_n} & = \mathbf{X_n}\mathbf{W'_q} + \mathbf{b'_q} \\
\mathbf{K_n} & = \mathbf{X_n}\mathbf{W'_k} + b'_k \\
\mathbf{V_n} & = \mathbf{X_n}\mathbf{W'_v} + b'_v \\
\text{Attention}(\mathbf{Q_n}, \mathbf{K_n}, \mathbf{V_n}) & = \text{softmax}(\frac{\mathbf{Q_n^T} \cdot \mathbf{K_n}}{\sqrt{d_k}})\mathbf{V_n}^T
\end{cases} \tag{3.2}
$$

where $\mathbf{X_n} \in \mathbb{R}^{1 \times n}$ is the numerical attributes value after normalization. $\mathbf{W'_q}, \mathbf{W'_k}, \mathbf{W'_v} \in \mathbb{R}^{n \times n}$ and $\mathbf{b'_q}, \mathbf{b'_k}, \mathbf{b'_v} \in \mathbb{R}^{1 \times n}$ are learning weight matrices. $1/\sqrt{d_k}$ is the scaling factor.

We concatenate the representations of categorical attributes and numeric attributes to construct the intermediate vector $\mathbf{C} \in \mathbb{R}^{1 \times (m \times d + n)}$.

$$
\begin{aligned}
\mathbf{C} = &\mathop{||}_{i=0}^{m} \text{Attention}(\mathbf{Q_{e_i}}, \mathbf{K_{e_i}}, \mathbf{V_{e_i}}) \\
&||\text{Attention}(\mathbf{Q_n}, \mathbf{K_n}, \mathbf{V_n})
\end{aligned} \tag{3.3}
$$

where || represents concatenation operation.

3.2.2 Graph Convolution Module

We construct an undirected graph by regarding each loan application as a node and adding weighted edges between nodes. Considering there are numerous attributes in raw data, we first select a subset of attributes to decrease the computational complexity. And then for every pair of nodes, we add an edge between them if they bear the same value on at least one attribute. We set the edge's weight to the number of attributes on which the two nodes share the same values.

To be specific, we define two thresholds, θ_m and θ_u, to select the attributes to participate in the graph construction. We choose an attribute if and only if 1) the proportion

of loan applications that have missing values on the attribute in all the applications is smaller than θ_m and 2) the number of distinct values for the attribute is fewer than θ_u. These two thresholds should be appropriately configured to avoid being excessively large (which introduces more noise) or small (which limits the auxiliary information usable to guide graph construction). We will study the impact of these thresholds in our experiments (Section 3.3.7).

Once the graph is constructed, we can apply graph convolution to aggregate neighbors' information and update the intermediate vector $\mathbf{C}$ [107]:

$$\mathbf{C}^{(l+\mathbf{1})} = \mathrm{ReLU}(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}\mathbf{C}^{(l)}\mathbf{W}^{(l)}) \tag{3.4}$$

where $\mathbf{C}^{(l+\mathbf{1})}$ is the intermediate vector after $l+1$ layers of graph convolution; $A \in \mathbb{R}^{N\times N}$ is the adjacency matrix for graph $\mathscr{G}$, $\tilde{A} = A + I$, and $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. $\tilde{D}$ is the degree matrix of $\tilde{A}$; $\mathbf{W}$ is the learned weight. We denote the last layer of graph convolution as $\mathbf{C^{(L)}}$

Finally, we define a λ to determine how much information to aggregate from neighbors:

$$\mathbf{G} = \lambda\mathbf{C} + (1-\lambda)\mathbf{C^{(L)}} \tag{3.5}$$

where $\mathbf{G}$ is the final representation of the intermediate vector; $\mathbf{C}, \mathbf{C^{(L)}}$ are the intermediate vector before and after graph convolution; λ is a learned parameter that controls how much raw information to remember.

3.2.3 Decoder Module

The decoder module (shown in the top-left of Fig. 3.2) uses two fully connected layers to reconstruct the input and to capture a better representation of raw data.

$$\tilde{\mathbf{V}} = \mathrm{ReLU}(\mathbf{W_{d_2}}\mathrm{ReLU}(\mathbf{W_{d_1}}\mathbf{G^T} + \mathbf{d_1}^\mathbf{T}) + \mathbf{d_2}^\mathbf{T}) \tag{3.6}$$

where $\tilde{\mathbf{V}}$ is the reconstructed inputs, $\mathbf{W_{d_1}}, \mathbf{W_{d_2}}, \mathbf{d_1}, \mathbf{d_2}$ are the weight matrix for linear transformation.

3.2.4 Classification Module

The classification module uses fully connected layers to make predictions on whether or not to authorize a loan (the top-right of Fig. 3.2 shows the detailed specification of the module, e.g., the number of layers and the number of neurons of each layer). Although

the intermediate vector $\mathbf{G}$ can capture semantic information from raw data (based on a stack of layers in our framework), it is prone to losing the shallow information in the original data. To make up for the information loss, we concatenate the original numeric input (i.e., a numeric vector) and the intermediate vector as the combined input for predicting the risk probability. The prediction result can be represented as:

$$\hat{\mathbf{y}} = \sigma(\mathbf{MLP}(\mathbf{G}||\mathbf{A_n})) \tag{3.7}$$

where σ denotes the sigmoid activation function.

3.2.5 Joint Learning

We apply a joint learning strategy for input vector reconstruction and credit default risk prediction. The loss function includes mean square loss and cross-entropy loss for the two learning tasks, respectively:

$$\begin{aligned} L = \frac{1}{|N|}\sum_{i\in N}(&-\alpha(y_i log\hat{y}_i + (1-y_i)log(1-\hat{y}_i)) \\ &+\beta\frac{1}{m+n}\sum_{j=1}^{m+n}(\mathbf{V_{ij}}-\tilde{\mathbf{V}}_{\mathbf{ij}})^2) + \lambda||\mathbf{\Theta}||_2 \end{aligned} \tag{3.8}$$

where $\alpha, \beta \in [0,1]$ balances the loss between the classification task and the reconstruction task. $y_i = 1$ indicates a positive case while $y_i = 0$ indicates otherwise. $\hat{y}_i$ is the predicted probability, i.e., the network's output after the softmax layer. $\mathbf{V_{ij}}, \tilde{\mathbf{V}}_{\mathbf{ij}}$ represent the original and reconstructed input vectors, respectively. $\mathbf{\Theta}$ denotes the set of trainable parameters. The last term is $L2$ regularization to mitigate overfitting.

3.3 Experiments

In this section, we report our experiments for evaluating our approach against several competitive baselines. Besides, we provide a further evaluation of our model under different configurations and parameter settings.

3.3.1 Datasets

We conduct experiments on two public datasets, which are representative of high-dimensionality and high-volume features of credit default risk data, respectively.

- **Home Credit Default Risk dataset**[2] covers various information about applicants, such as family information, income and expenditure, credit records, loan records, and repayment history. In contrast to most credit default risk datasets, the Home Credit Default dataset emphasizes installment lending, specifically targeting individuals with limited or no credit history. Given these insufficient or non-existent credit histories, financial institutions must rely on alternative data sources,Äîsuch as telecommunications information, transactional records, and payment histories,Äîto assess creditworthiness and provide secure lending services. Consequently, the application data are complex, multi-source, and heterogeneous. More concretely, there are 307,511 train samples and 48,744 test samples, each having 477 numeric attributes and 55 categorical attributes in the raw data. The ratio of positive to negative samples in the train set is approximately 1:11. Values are missing for half of those attributes. We randomly draw 10% of the training data as the validation set.
- **Lending Club dataset**[3] contains millions of loan records with 151 attributes from 2007 to 2018. Following previous work [4], we remove meaningless attributes (e.g., URL, member_id), together with those attributes with more than 30% missing values, and then fill the remaining missing values with zeros. Each sample has 7 categorical attributes and 16 numeric attributes after preprocessing. The ratio of positive to negative samples is approximately 1:4. We use the most recent 10% records for testing and the rest for training.

3.3.2 Baseline Methods and Evaluation Metric

We select several recent competitive methods, which reflect the state-of-the-art research, to compare with our approach:

- **Logistic Regression [178]**: a simple and efficient linear model for binary classification. It is widely used in many applications, including machine learning, medical fields, and social sciences [178].
- **Decision Tree [313]**: a technique that classifies samples following an ordering of attributes with a tree structure, where each internal node represents a test on an attribute, each branch denotes the outcome of this test, and each leaf node signifies

[2] https://www.kaggle.com/c/home-credit-default-risk/overview
[3] https://www.kaggle.com/wordsforthewise/lending-club

the class label associated with a particular sample. The paths from the root to each leaf node represent distinct classification rules [313].

- **Random Forest [178]**: an ensemble learning method that trains a multitude of decision trees and determines the label via majority voting.
- **XGBoost [4]**: a decision-tree-based ensemble machine learning algorithm that uses a gradient boosting framework.
- **Fully Connected Deep Network [4]**: a network with fully connected layers. We choose ReLU as the activation function and cross-entropy loss for optimization.
- **Convolution Neural Network [311]**: a network that feeds the concatenation of the representation of categorical data (obtained by convolution operations) and numeric attributes to fully connected layers for making predictions.
- **Wide & Deep Neural Network [38]** applies a linear model to improve the sparsity of categorical features and robustness of models via cross-product feature transformations. For the deep module, a feed-forward neural network is applied for feature representation, while the wide module endeavors to improve the memory of the network.

For SaM-GNN, we set the embedding size $E = 5$, $\alpha = 0.5, \beta = 0.5$ for the loss function. For graph construction, we set $\theta_m = 0.3$ and $\theta_u = 20$ for attributes selection. The learning rate of the Adam optimizer is initialized to 0.001, which decays by 0.1 after every 50 epochs. Batch size and L2 penalty are set to 500 and 10^{-5}, respectively. For the classification module in SaM-GNN, we apply cross-entropy loss, use ReLU as the activation function, and set the dropout rate to 0.35 for all the fully connected layers except the last. The number of neurons in the layers is 512, 256, 128, 64, 32, 2 for the Home Credit Default Risk dataset and 32, 16, 16, 2 for the Lending Club dataset.

For Logistic Regression, we set the inverse of the regularization strength to $C = 10^{-4}$. For the Decision Tree, the maximum tree depth is fixed at 5. For Random Forest, we set the number of estimators to $1,000$. In XGBoost, we use $10,000$ gradient-boosted trees with a subsample ratio of 0.5, and the learning objective is logistic regression. We implement an early-stopping strategy, with the stopping round set to 100. All other hyperparameters for these baseline methods remain at their default values.

In the Convolutional Neural Network (CNN), the number of input channels C_{in} and output channels C_{out} are set to 1 and 3, respectively. The kernel size is set to $H \times W$ where W denotes the embedding size, which we set to 5, and $H \in \mathscr{H} = \{2,3,5,10,15,25,40,50\}$.

We apply max pooling for each feature after convolution and concatenate the results. Next, we concatenate the numerical attributes, and the fully connected layers contain 256, 128, 64, 32, and 2 neurons, respectively.

For the Wide & Deep Neural Network, we apply a cross-product transformation in the wide component to attributes with values below 5 and use a sigmoid activation function. In the deep component, the numeric and categorical attributes are consistent with those used in SAGM. We set the embedding size for categorical features to 5, concatenate the embeddings with the numeric features, and feed this vector into five ReLU layers. Finally, the outputs of the wide and deep components are combined using a weighted sum of their log odds to produce the prediction. The remaining hyperparameters for the Fully Connected Deep Network, Convolutional Neural Network, and Wide & Deep Neural Network are consistent with those used in SAGM.

Following previous studies [2], we use *Area under the ROC Curve (AUC)* as the evaluation metric to evaluate models' performance. AUC has the characteristic of signifying the probability that positive samples receive higher scores than negative samples, revealing the classification model's ability to rank samples. Therefore, it can effectively reduce false alarms (or false positive rates) and decrease potential financial loss, making it especially suitable for the credit default risk prediction problem.

3.3.3 Comparisons with Baselines and Ablation Study

To demonstrate the overall performance of the deep neural network, we compare it with different baselines for credit default risk prediction. Table 3.1 shows the performance (with respect to AUC) of different methods. Generally, deep neural networks and ensemble methods outperform traditional models (Decision Tree, Logistic Regression), which have limited feature representation ability for high-dimensional data, while Boosting methods (XGBoost) outperform the Bagging method (Random Forest), and both of them are significantly superior to the decision tree (i.e., base learner). However, the training speed of XGBoost is relatively slower than that of Random Forest, as the bagging strategy enables parallel ensemble processing, whereas boosting, used by XGBoost, operates as a sequential ensemble method. Neural network-based models generally perform better than shallow models (traditional models, Bagging method) thanks to their strong feature representation and non-linear learning ability. Regarding the Home Credit Default Risk dataset, SaM-GNN outperforms all the other methods. XGBoost and Random Forest achieve a remarkable improvement after incorporating the intermediate vector generated by SaM-GNN, demonstrating the effectiveness of SaM-GNN in improving existing

methods. As for the Lending Club dataset, SaM-GNN and its variant without the Decoder module achieve a significant improvement (over 26%) over most of the other methods; they outperform the third-best method (i.e., XGBoost + Intermediate Vector) by a large margin of 0.1 in AUC, which reconfirms the superiority of our approach.

While our proposed model demonstrates effectiveness in credit risk prediction, it also faces several practical challenges. First, the integration of self-attention, graph convolution, and decoder/classification modules inevitably increases the model's complexity. To address this, we adopt parameter sharing and dimensionality reduction techniques to control the number of parameters. Second, the computational cost of graph-based deep models can be substantial, particularly for large-scale datasets. In future work, we may explore more efficient GNN variants to reduce training overhead further. Third, constructing meaningful graphs from tabular data remains a non-trivial problem. In our work, attribute similarity analysis guides the graph construction process, while in Chapter 4, we incorporate adaptive or data-driven graph learning methods to improve generalization and robustness.

Our ablation study (based on comparisons between SaM-GNN and its variants, as shown in Table 3.1) demonstrates the effectiveness of considering the auxiliary information, i.e., similar credit application records, via graph convolution. While the Decoder module avails SaM-GNN's performance on Home Credit Default Risk dataset, it does not significantly impact the performance on Lending Club dataset—SaM-GNN obtains similar results regardless of whether it incorporates the Decoder module; this suggests the Decoder module is more effective on challenging tasks than easy ones—the classification task on Lending Club Dataset is not liable to overfit even without the Decoder module, given that the dataset contains millions of loan records but only 23 attributes.

3.3.4 Impact of Sampling Methods

Our datasets have imbalanced distributions over classes, with the ratios of positive to negative samples being 1:11 and 1:4 for Home Credit Default Risk and Lending Club datasets, respectively. Therefore, we test the effectiveness of three sampling strategies [4] in overcoming the class imbalance issue:

- **Upsampling**: Upsampling the positive samples and balancing the data distribution. Let δ_{up} as the desired ratio of the number of samples in the minority class over the number of samples in the majority class after resampling.

Table 3.1: Performance (AUC) of different models. The best results are highlighted in boldface. ↑ and ↓ denote improvement and drop in performance, respectively. The numbers besides the up/down arrows indicate the percentages by which the models improve their original versions.

Method	Home Credit	Lending Club
Logistic Regression	0.71739	0.69017
Decision Tree	0.72383	0.69925
Random Forest	0.74657	0.70380
XGBoost	0.76869	0.71616
Fully Connected Neural Network	0.76908	0.70448
Convolution Neural Network	0.77070	0.70961
Wide & Deep Neural Network	0.75824	0.70207
SaM-GNN	**0.78605**	**0.96982**
SaM-GNN w/o Decoder Module	0.77395 (↓ 1.54%)	0.96347 (↓ 0.65%)
SaM-GNN w/o Decoder & Graph Convolution	0.77026 (↓ 2.01%)	0.71253 (↓ 26.05%)
Random Forest + Intermediate Vector	0.75270 (↑ 0.82%)	0.86833 (↑ 23.38%)
XGBoost + Intermediate Vector	0.77289 (↑ 0.55%)	0.87326 (↑ 21.94%)

- **Downsampling**: Downsampling the negative samples and balancing the data distribution. Let δ_{down} as the ratio of the number of negative samples to the original number after resampling.

- **SMOTE [18]**: Selecting k nearest neighbors in the feature space for the minority class samples, drawing a line between the neighbors in the feature space, and drawing a new sample at a point along that line. Let δ_{smote} as the desired ratio of the number of samples in the minority class over the number of samples in the majority class after resampling.

Specifically, we investigate the performance of SaM-GNN under varying δ, i.e., $\delta \in \{0.2, 0.3, 0.5, 0.6, 0.8, 1.0\}$ (the ratio of positive to negative samples after resampling). The other hyperparameters remain the same as SAGM. Our results (Table 3.2) suggest that among the three sampling methods, *upsampling* consistently results in the best AUC on both datasets. Also, for both datasets, the performance of sampling methods tends to fluctuate under varying values of the ratio (δ), indicating the best configurations of δ should be determined empirically rather than by following certain rules.

Table 3.2: Performance (AUC) of SaM-GNN under different sampling strategies. δ is the ratio of positive to negative samples after resampling. The best result under each ratio setting is highlighted in boldface.

Dataset	Home Credit			Lending Club		
Ratio (δ)	0.5	0.75	1.0	0.5	0.75	1.0
Upsamping	**0.78665**	**0.78529**	0.76584	**0.96390**	**0.95482**	**0.95394**
Downsampling	0.77398	0.77425	**0.77825**	0.93928	0.91154	0.89367
SMOTE	0.76134	0.76130	0.76307	0.93250	0.95238	0.94642

3.3.5 Impact of Vector-fusion Methods

Vector-fusion methods are for fusing the feature representations of categorical and numeric inputs to construct the intermediate vector. We study the impact of two commonly used vector-fusion methods, *concatenation* and *mean-pooling*, on the performance of SaM-GNN.

Our results (Figure 3.3) show that *concatenation* (i.e., what we use in SaM-GNN) generally leads to better performance than *mean-pooling*—while both methods result in similar results on the Home Credit dataset, *concatenation* consistently outperforms *mean-pooling* on the Lending Club dataset. *Mean-pooling* tends to favor larger embedding sizes, obtaining two out of the top-three best results under embedding-size=50 and 100 on the datasets—larger embedding sizes help improve the network's feature representation ability when *mean-pooling* is applied. The optimal embedding size for *concatenation* is more data-specific. Specifically, *concatenation* requires smaller embedding sizes (e.g., 5, 15, 20) for smaller datasets (e.g., Home Credit) yet larger embedding sizes (e.g., 20, 50, 100) for larger datasets (e.g., Lending Club) to deliver the best results. This makes sense as Lending Club contains more training samples with lower dimensionality, allowing for a 'wider' neural network to excel without overfitting.

3.3.6 Impact of Multi-task Learning Parameters

We study the performance of our model (i.e., SaM-GNN without the graph convolution module) under varying parameters for multi-task learning α and β while keeping the other hyperparameters the same as SaM-GNN. The results on the Home Credit Default Risk dataset (Table 3.3) show that the AUC values above the diagonal are generally greater than below, indicating it improves the performance to force the network to pay more attention to obtaining a better intermediate vector representation. We omit to show

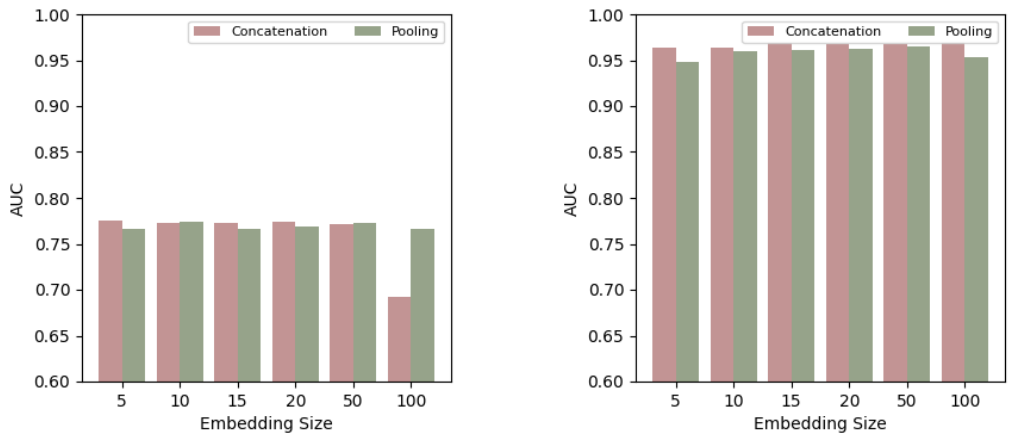


Figure 3.3: Performance (AUC) of SaM-GNN with different vector-fusion methods on Home Credit (left) and Lending Club (right) Datasets.

the results on the Lending Club dataset, on which the impact of multi-task learning is less evident.

Table 3.3: Performance (AUC) of SaM-GNN (without considering the graph convolution module) under different parameter settings of multi-task learning on the Home Credit Default Risk dataset. The best result in each row/column is highlighted in boldface.

α \ β	0.0	0.2	0.4	0.6	0.8	1.0
0.2	0.74548	0.76145	**0.77629**	0.76835	0.76600	0.75980
0.4	0.75684	0.76086	0.76550	0.76078	0.75654	**0.76702**
0.6	0.74604	0.75617	0.76626	**0.77077**	0.76352	0.76245
0.8	0.75136	0.75099	0.75422	0.76565	**0.76814**	0.76579
1.0	**0.76426**	**0.77107**	0.75140	0.76066	0.75738	**0.77991**

3.3.7 Impact of Graph Construction Parameters

We further study the impact of θ_m and θ_u, which affect attribute selection during graph construction and, in turn, determine the final graph structure. To this end, we test the performance of SaM-GNN under $\theta_u \in \{20, 30, 50, 100, 200, 500, 1000\}$ for Home Credit dataset, $\theta_u \in \{5, 10, 20, 30, 50\}$ for Lending Club dataset, and $\theta_m \in \{0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$. Intuitively, the graph contains more edges under smaller values of θ_u and θ_m. Our results show our model favors denser graphs derived from the datasets—SaM-GNN achieves

the best AUC (0.78002 and 0.96652) under the smallest θ_u values (20 and 5) for Home Credit and Lending Club datasets, respectively. θ_m has a slighter impact on the results when compared with θ_u.

3.4 Conclusion

In summary, we propose a self-attention graph neural network with multi-task learning for credit default risk prediction. The network features self-attention and graph convolution to represent heterogeneous data with missing values, along with multi-task learning of classification and decoder modules for model training. Extensive experiments on two real credit default risk prediction datasets demonstrate the superiority of our approach to existing models. Besides, feature representations from our approach help improve the performance of Bagging and Boosting methods.

Although our proposed graph neural network for credit risk prediction achieved optimal results compared to existing works, the graph structure modeling requires more computing cost and memory, which may limit the model's applicability to very large datasets. In addition, with more connections, the model is more likely to fit noise rather than meaningful patterns, potentially reducing its generalization ability, making the model harder to interpret, a concern that is especially important in financial applications. Moreover, the model's performance depends on careful tuning of hyperparameters, such as θ_u and θ_m, and ensuring robustness across different datasets can be challenging.

CHAPTER 4

MULTI-VIEW GNNS FOR LOAN DEFAULT RISK PREDICTION

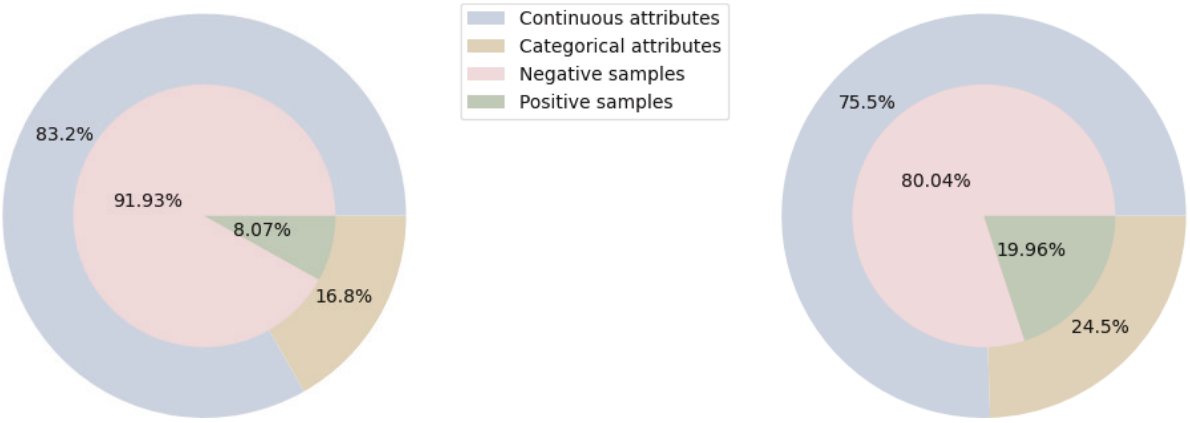
4.1 Introduction[1]

The experimental results in Chapter 3 demonstrated the powerful capability and flexibility of graph neural networks in loan default risk prediction. Nevertheless, there remain two key challenges unaddressed:

- **Missing Values and Imbalanced Data Distribution.** Due to information sensitivity, privacy requirements, and trust issues, users' information may be incomplete or even non-existent. For instance, the percentage of records containing missing values exceeds 50% in the Home Credit Default Risk dataset. In addition, the default samples only account for a small part of most of the datasets. As shown in Figure 4.1, as the default records (negative) are small samples in practical scenarios, the data distribution is imbalanced, which may lead to the unequal cost of misclassification errors and scarce accuracy of prediction [40, 307]. In general, the missing values and skewed data distribution pose a huge challenge for default risk prediction.

- **Explicit Connections between Records.** Deep learning methods focus more on record representation learning while neglecting the explicit connection re-

[1]This Chapter is based on our published work: Multi-view GCN for loan default risk prediction.



(a) Home Credit Dataset (b) Lending Club Dataset

Figure 4.1: The percentage of categorical and continuous attributes, positive and negative samples account for all the records on the Home Credit Default Risk Dataset and Lending Club Dataset.

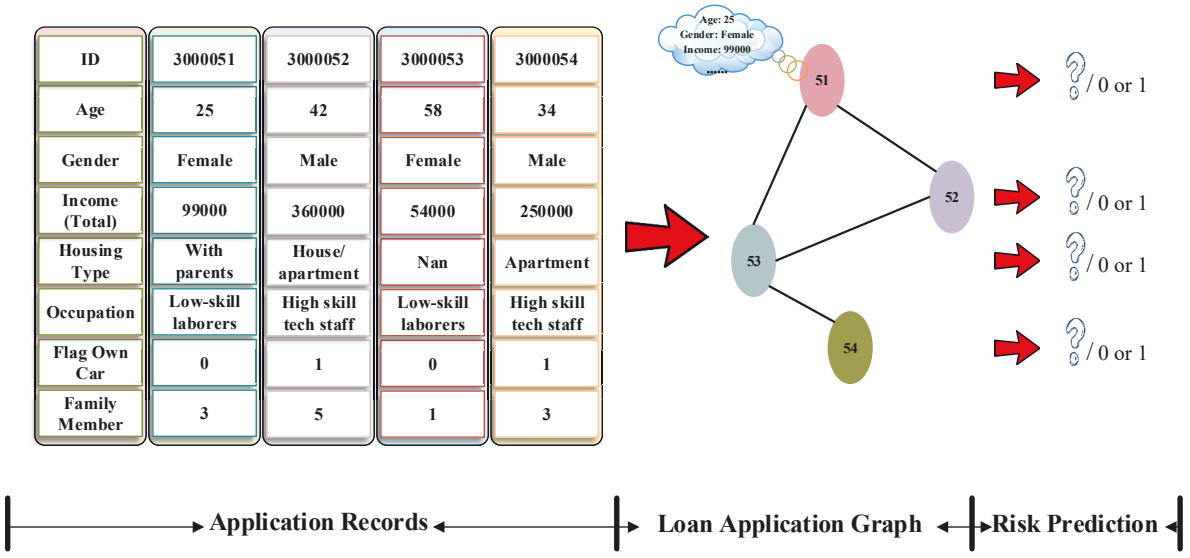


Figure 4.2: A toy example of loan graph. Each loan application record (ID) can be represented as a node. Based on the similarity, we could add edges between nodes.

lationships between records. We believe the similar information encapsulated in the records could also facilitate the feature representation and final prediction [89, 227, 249].

As shown in Figure 4.2, the record in the table can be regarded as a node. Accordingly, a loan application graph could be constructed via the similarity among each record. Through node classification, the loan default risk problem could be solved effectively.

To this end, we propose a Multi-view Graph Convolution Network (MGCN) for loan default prediction. First, we endeavor to construct multi-view loan application graphs via similarity calculation for small sample augmentation and skew distribution adjustment. Besides, based on graph convolution, similar loan application records can be aggregated as auxiliary information, thus, the model's robustness to missing values could be improved. We make the following contributions in this Chapter:

- We devise multi-view loan application graphs (MLAGs) via the similarity between application records. By adjusting the thresholds of similarity, we could control the sparsity of graphs for multi-view graph construction. Then, the data augmentation strategy is applied to small samples to balance the data distribution.
- We propose a multi-view graph convolution network (MGCN) for loan default prediction. By information aggregation and propagation, similar record information encapsulated in the graph structures can be introduced as auxiliary information to alleviate the issue of missing values.
- We conducted experiments on three real-world open loan default risk prediction datasets. The experimental results show a significant performance improvement in our approach over state-of-the-art methods [71, 227, 249].

4.2 Problem Formulation

Based on the raw application records, multi-view loan application graphs (MLAGs) will be constructed as follows.

4.2.1 Loan Application Graph (LAG)

Given a set of loan application records $\mathscr{D}$, we denote an application record as r and $r \in \mathscr{D}$. In LAG, the loan application record r will be represented by a node v. And the corresponding attributes of v are denoted as x_v. Let $\mathscr{X}^{\mathcal{V}} = \{x^v \mid v \in \mathcal{V}\}$ as the set of attributes, which can be further classified as categorical attributes ${\mathscr{X}^{\mathcal{V}}}_{Ca}$ (e.g., gender, education, occupation, etc.) and continuous attributes ${\mathscr{X}^{\mathcal{V}}}_{Co}$ (e.g., age, income, etc.) based on the number of unique values. Specifically, in this Chapter, we define the attribute whose containing number of unique values is less than the threshold as a categorical attribute; otherwise, it belongs to the continuous attribute. It can be formalized as follows:

$$\begin{cases} x \in \mathscr{X}^{Ca} & if\ \#(x) < \eta N \\ x \in \mathscr{X}^{Co} & \text{others} \end{cases} \tag{4.1}$$

where $\mathscr{X}^{Ca}, \mathscr{X}^{Co}$ are categorical attributes set and continuous attributes set, respectively. N is the number of total records in this dataset. η is a hyperparameter, which controls the granularity of attributes modeling, i.e., if the η is bigger, more attributes will be encoded to category embedding.. We set $\eta = 10^{-4}$. $\#(\cdot)$ is the counting operation. Hence, for attribute x, if the number of containing values is less than threshold ηN, we categorize x as a categorical attribute; otherwise, x belongs to a continuous attribute.

Assuming each attribute includes m categorical attributes and n continuous attributes. We, thereby, calculate the similarity between any two nodes. If the similarity between node v_i and node v_j is over the threshold δ, we add an edge e_{ij} between them. Consequently, a loan application graph (LAG) $\mathscr{G} = \{\mathscr{V}, \mathscr{E}, \mathscr{X}^{\mathscr{V}}\}$ could be constructed, where $\mathscr{V}$ and $\mathscr{E}$ represent the set of nodes and edges in the LAG, respectively.

4.2.2 Multi-view Loan Application Graphs (MLAGs)

As described in Chapter 4.2.1, by changing the similarity threshold δ, we could adjust the structure of LAG dynamically. Therefore, given a series of $\delta \in \Delta, \Delta = \{\delta \mid 0 < \delta < 1\}$, multi LAGs can be constructed. We denote the $\mathbf{G} = \{\mathscr{G}_1, \mathscr{G}_2, ...\}$ as the multi-view loan application graphs (MLAGs). The different LAGs allow us to dynamically select and aggregate information from similar records in a flexible fashion. Besides, the MLAGs will also be utilized for small sample argumentation.

4.2.3 Loan Credit Default Risk Prediction

Given $\mathscr{D} = \{r, y_r\}$, where $y_r \in \{0, 1\}$ is the label of loan default and r is users' correspondent application information and historical records. Based on the application record r, the task of Loan credit default risk prediction aims to identify the likelihood that the user will default or not.

4.3 Methodology

We first give an overall illustration of our proposed MGCN model. Then, we introduce each module of MGCN specifically. Figure 4.3 illustrates the MGCN structure, which

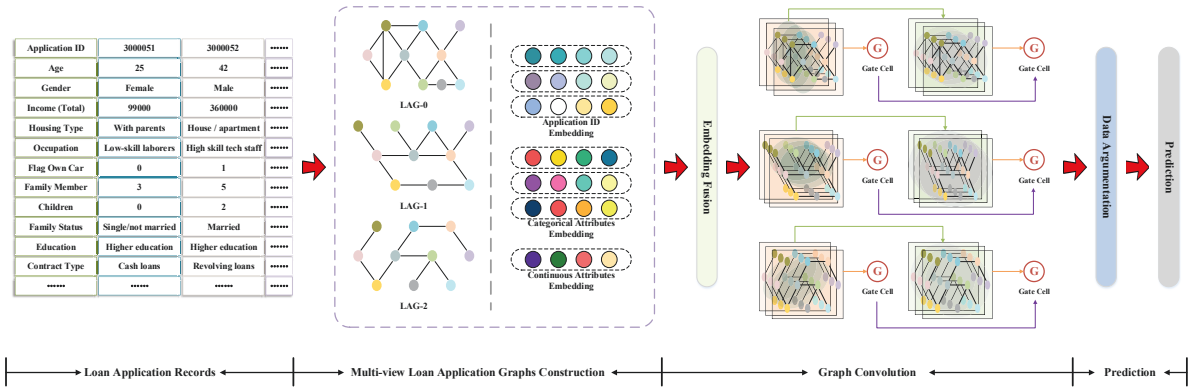


Figure 4.3: The architecture of MGCN. It includes four main parts, which are arranged from left to right: 1) Multi-view Loan Application Graph Construction; 2) Graph Convolution Layer; 3) Data Argumentation Layer; and 4) Prediction Layer.

contains four parts:

- **MLAGs Construction.** Based on the Chapter 4.2.1 and 4.2.2, each loan application record r can be regarded as a node v. Therefore, by calculating the similarity between any two nodes and adjusting the thresholds δ, we could construct the MLAGs for loan default risk prediction.

- **Node Embedding and Graph Convolution.** We concatenate the category attributes embedding and continuous attributes embedding for application record (i.e., node) representation generation. Then, attributed to the information propagation and aggregation from similar neighbors, a graph convolution with a gate mechanism is proposed to alleviate the issue of missing values.

- **Data Augmentation.** As the data distribution is imbalanced in a practical scenario, for each negative (small) sample, we collect all the representations from MLAGs, while for each positive sample, we fuse all the representations under different LAGs with a soft attention mechanism to generate a single representation for data distribution re-balance.

- **Prediction Layer.** After obtaining the loan application representation updated by the graph convolution layer, we devise a prediction layer (a feed-forward network) for loan default risk prediction.

4.3.1 Multi-view Loan Application Graphs Construction

As aforementioned, there are two category attributes for each loan application record. Hence, we calculate the similarity for the two kinds of attributes, respectively, for loan application graph construction. To be specific, for categorical attributes x^{Ca}, we calculate the Jaccard similarity between any two records, which can be formalized as follows:

$$\mathrm{Sim}_{Ca}(x_i^{Ca}, x_j^{Ca}) = \frac{x_i^{Ca} \cap x_j^{Ca}}{x_i^{Ca} \cup x_j^{Ca}} \tag{4.2}$$

where x_i^{Ca}, x_j^{Ca} are categorical attributes values of record r_i and r_j, respectively. Hence, the numerator counts the number of attributes with the same values for records r_i and r_j. And the denominator counts the total number of attribute values from r_i and r_j. n is the total number of attributes.

For continuous attributes, we utilize Euclidean distance to measure the similarity between each record.

$$\mathrm{Sim}_{Co}(x_i^{Co}, x_j^{Co}) = 1 - \sum_{k=1}^{n} (x_{i,k}^{Co} - x_{j,k}^{Co})^2 \tag{4.3}$$

where $x_{i,k}^{Co}, x_{j,k}^{Co}$ are k-th continuous attribute values (undergo normalization operation) of records r_i and r_j, respecitvely.

Thus, the final similarity between records r_i and r_j can be defined as follows:

$$\mathrm{Sim}(r_i, r_j) = \alpha \cdot \mathrm{Sim}_{Ca}(x_i^{Ca}, x_j^{Ca}) + (1-\alpha) \cdot \mathrm{Sim}^{Co}(x_i^{Co}, x_j^{Co}) \tag{4.4}$$

where α is a hyperparameter to balance the similarity between categorical attributes and continuous attributes. As shown in Figure 4.1, the categorical attributes account for the major part of all the attributes, and it is sufficient to measure the similarity between any two nodes via categorical attributes, thus, we only calculate the similarity of categorical attributes to reduce the computation complexity of MLAGs construction.

Again, we further define a hyperparameter δ, if $\mathrm{Sim}(r_i, r_j) > \delta$, we add an edge between node v_i and v_j. Hence, the weight of edge $e_{i,j}$ is $\mathrm{Sim}(r_i, r_j)$. Adjusting the threshold δ, we could control the sparsity of the graph structure.

4.3.2 Embedding Layer

4.3.2.1 Categorical Attributes Embedding

Firstly, we project the categorical attributes of each record to a latent space through an embedding table $\mathbf{E} \in \mathbb{R}^{M \times d}$. Here, M is the total number of unique values for all the categorical attributes, and d is the embedding dimension. The embedding operation is implemented as follows,

$$\mathbf{x_v^{Ca}} = \text{Onehot}(x_v^{Ca})\mathbf{E} \tag{4.5}$$

where Onehot$(\cdot)$ is one-hot operation, which encodes the original categorical attributes x_v^{Ca} of node v to an one-hot vector. Onehot$(x_v^{Ca}) \in \mathbb{R}^{m \times M}$, m is the number of categorical attributes for each record.

4.3.2.2 Continuous Attributes Embedding

For each continuous attribute, we adopt Norm$(\cdot)$ operation for normalization, which will further undergo a linear projection matrix for representation initialization.

$$\mathbf{x_v^{Co}} = \text{Norm}(\mathbf{x_v^{Co}})\mathbf{W} \tag{4.6}$$

where $\mathbf{W} \in \mathbb{R}^{n \times n}$ is a learnable matrix.

4.3.2.3 Node Representation

We flatten the categorical attributes embedding and concatenate the continuous attribute representation to construct the node representation $\mathbf{x_v}$. $\mathbf{x_v} \in \mathbf{X}$, and $\mathbf{X} \in \mathbb{R}^{|V| \times (m \times d + n)}$.

$$\mathbf{x^v} = \mathop{\Big\|}_{k=0}^{m} \mathbf{x_v^{Ca_k}} \,||\, \mathbf{x_v^{Co}} \tag{4.7}$$

where $\mathbf{x_v^{Ca_k}} \in \mathbb{R}^d$ is k-th categorical attribute embedding of node v. $x_v^{Co} \in \mathbb{R}^n$ is the continuous attributes embedding. $||$ is the concatenation operation.

4.3.3 Graph Convolution Layer

We utilize graph convolution on each LAG for information propagation and aggregation [107]. Thus, similar records can be introduced as auxiliary information to alleviate

the missing value issue. However, the deeper graph convolution operation will incur the over-smoothing [127] (also known as the information loss [163] problem). Consequently, we propose to aggregate the information from the shallow graph convolution layer via a gate mechanism for node representation updating. For LAG - g, the node representation can be formulated as:

$$
\begin{aligned}
\mathbf{X_g^{(l)}} &= \mathrm{ReLU}(\tilde{\mathbf{D}}_\mathbf{g}^{-\frac{1}{2}}\tilde{\mathbf{A}}_\mathbf{g}\tilde{\mathbf{D}}_\mathbf{g}^{-\frac{1}{2}}\mathbf{X_g^{(l-1)}}\mathbf{W^{(l-1)}}) \\
\mathbf{H_g^{(l)}} &= \alpha\mathbf{X_g^{(l)}} + (1-\alpha)\mathbf{H_g^{(l-1)}} \\
\alpha &= \mathrm{Diag}(\sigma(\mathbf{X_g^{(l)}}\mathbf{W_X} \cdot \mathbf{H_g^{(l-1)}}\mathbf{W_H}))
\end{aligned}
\tag{4.8}
$$

where $\mathbf{X_g^{(l)}}$ is the node representation after l-h graph convolution layer with regard to g. $\mathscr{A}_g \in \mathbb{R}^{N\times N}$ is the adjacency matrix of graph g, $\tilde{\mathbf{A}}_\mathbf{g} = \mathbf{A_g} + I$, and $\tilde{\mathbf{D}}_{\mathbf{g_{ii}}} = \sum_j \tilde{\mathbf{A}}_{\mathbf{g_{ij}}}$. $\tilde{\mathbf{D}}_\mathbf{g}$ is a degree matrix of $\tilde{\mathbf{A}}_\mathbf{g}$; $\mathbf{W^{l-1}}, \mathbf{W_X}, \mathbf{W_H}$ are learnable matrices; ReLU$(\cdot)$ and $\sigma(\cdot)$ (i.e., sigmoid$(\cdot)$) are activation functions; Diag$(\cdot)$ means obtain diagonal values. In this Chapter, we initialize $\mathbf{X_g^{(0)}}, \mathbf{H_g}^{(0)}$ as the node embedding generated by secion 4.3.2.3 and obtain the last update $\mathbf{H_g^L}$ as the graph convolution output of g.

4.3.4 Data Augmentation Layer

As the default records (i.e., negative samples) are small samples in practical scenarios, the data distribution is skewed. Hence, we carry out data augmentation on small samples to balance the data distribution. To be specific, for negative samples, we collect all the representations from different LAGs. So as to the positive sample, considering multi-view graph convolutional networks (MGCNs) can indeed face optimization challenges, especially when different views are inconsistent or provide conflicting learning signals. Accordingly, in the implementation, we employed view-specific normalization and adaptive weighting mechanisms to balance contributions from different views, reducing conflicts between them. Specifically, we mix the different representations from LAGs with a soft-attention mechanism as below.

$$
\begin{aligned}
\mathbf{h^p} &= \sum_{g=1}^{s} \alpha_g(\mathbf{h_g^p}\mathbf{W_p}) \\
\alpha_g &= \frac{\mathbf{x^p}\mathbf{W_p} \cdot \mathbf{h_g^p}\mathbf{W_p}}{\sum_{g=1}^{s} \mathbf{x^p}\mathbf{W_p} \cdot \mathbf{h_g^p}\mathbf{W_p}}
\end{aligned}
\tag{4.9}
$$

where $\mathbf{x^p}$ is the positive sample's embedding; $\mathbf{h_g^p}$ is the positive sample's representation with regard to LAG-g after graph convolution derived from Equation (6.8). $\mathbf{W_p}$ is a learnable matrix. s is the number of LAGs.

4.3.5 Prediction Layer

Note that each loan application record r can be recognized as a node v in the MLAGs. Therefore, the loan default risk prediction can be transformed into the node classification task. Specifically, we implement the feed-forward network (FFN) and with *softmax* function for node classification, which is formulated as follows:

$$\begin{aligned} \hat{\mathbf{y}} &= \mathrm{ReLU}(\mathbf{W_2} \cdot \mathrm{ReLU}(\mathbf{W_1} \cdot \mathbf{h^L} + \mathbf{b_1}) + \mathbf{b_2} \\ \hat{\mathbf{y}} &= \mathrm{softmax}(\hat{\mathbf{y}}) \end{aligned} \tag{4.10}$$

where $\mathbf{h^L}$ is the record representation after data augmentation. $\hat{\mathbf{y}}$ is the probabilities of predicted labels. $\mathbf{W_1}, \mathbf{W_2}, \mathbf{b_1}, \mathbf{b_2}$ are the trainable weight matrices and the bias vectors , respectively.

4.3.6 Model Learning

The cross-entropy with L2 regularization is adopted as the loss function to train the MGCN in an end-to-end mode. Formally, we define the loss function over all the loan application records $\mathscr{D}$ as:

$$\mathscr{L}(\mathbf{\Theta}) = -\sum_{<r,y>\in\mathscr{D}} y \log(\hat{\mathbf{y}}) - \lambda ||\mathbf{\Theta}||_2^2 \tag{4.11}$$

where y is the ground-truth of loan risk prediction, $\mathbf{\Theta}$ is the trainable parameters of our proposed model, and λ is a regularized hyperparameter.

4.4 Experiments

We investigate the effectiveness of our designed model. We conduct extensive experiments on three real-world public datasets. First, we describe the experimental settings, including the datasets, methods in comparison, evaluation metrics, and experimental setup. Then, we discuss the performance comparison and results from the model analysis. Additionally, we perform ablation tests to demonstrate the effectiveness of every component in our model. We aim to answer the following research questions:

Table 4.1: The Statistical information of datasets. Label-1 means the loan default samples.

Statistic Index	Home Credit	Lending Club	PPD
# Total	307,511	2,260,701	30,000
# Label-1	24,825	268,559	2,198
# Label-0	282,686	1,076,751	27,802
# Continuous Attributes	288	113	208
# Categorical Attributes	58	38	20
Ratio of missing values	50%	100%	21%

- **RQ1:** Does our model outperform the state-of-the-art loan default risk prediction methods on the real-world datasets?
- **RQ2:** How do the key components (e.g., gate mechanism in graph convolution layer and multi-view loan application graphs) and hyperparameters of our model benefit the prediction?
- **RQ3:** How about sampling strategies (e.g., up and down-sampling, SMOTE method) to MGCN performance improvement compared with our data augmentation?

4.4.1 Dataset

We evaluate our method performance on three public datasets. Table 4.1 summarizes the basic statistics of these datasets. Observing the table 4.1, we could find that 1) the ratio of loan default samples to others is up to $1:12.6$, the data distribution is skewed significantly; 2) the number of continuous attributes is about 2.97 to 10.4 times that of categorical attributes; 3) a large number of missing values, e.g., for Lending Club dataset, all the attributes contain missing values.

- **Home Credit Default Risk**[2] primarily to lend to those people with little or no credit history. Due to incomplete or non-existent credit histories, financial institutions are required to evaluate various alternative data for a positive and safe borrowing service. Hence, the application records are always complex, multi-source, and heterogeneous. This dataset covers various information about applicants, e.g., family information, income and expenditure, credit records, loan records, repayment history, etc.

[2] `https://www.kaggle.com/c/home-credit-default-risk/overview`

- **Lending Club**[3] is a P2P lending dataset, which contains millions of loan records with 151 attributes from 2007 to 2018. All the attributes in raw data have missing values except *id*. Following previous work [4], we remove meaningless attributes (e.g., URL, member_id) and the attributes with more than 30% missing values. Afterwards, we fill in the missing values with zeros.

- **PPD**[4] is a public loan default prediction dataset released from a risk control algorithm competition at the Heywhale community. Each record comprises 208 continuous attributes and 20 categorical attributes, which cover the basic information of lenders, the login information, and modification records.

4.4.2 Experimental Settings

4.4.2.1 Evaluation Metrics

Following previous works [89, 147, 306], we select Recall, F1 score, and AUC (i.e., Area Under the ROC Curve) to evaluate the performance of each model on the above three public datasets.

AUC signifies the probability that the positive item sample‚Äôs score is higher than the negative item sample‚Äôs score, a larger AUC indicates a better performance. Compared with precision, recall, and F1 value, AUC trends reveal the rate of false positives (also known as false alarms). Considering the cost-sensitive and imbalanced distribution in the default risk prediction task, it may be more reasonable to calculate AUC for model evaluation, which is defined as:

$$AUC = \frac{\sum_{r \in \mathscr{R}^+} rank_{r-} \frac{|\mathscr{R}^+| \times (|\mathscr{R}^+|+1)}{2}}{|\mathscr{R}^+| \times |\mathscr{R}^-|} \tag{4.12}$$

where $\mathscr{R}^+$ and $\mathscr{R}^-$ denote the positive and negative sets in the testing set, respectively. And $rank_r$ indicates the rank of record r via the score of prediction.

4.4.2.2 Compared Methods

We compare our model with multiple representative and competitive methods from four categories: machine learning methods (e.g., Logistic Regression, Decision Tree, Random Forest, XGBoost), deep learning methods (e.g., DNN, CNN, Wide & Deep Neural

[3] https://www.kaggle.com/wordsforthewise/lending-club

[4] https://www.heywhale.com/home/competition/56cd5f02b89b5bd026cb39c9/content/1

Network), unsupervised graph embedding methods (e.g., DeepWalk, Node2vec), and GCN-based methods (e.g., GCN, GraphSage, GAT, SGC, GCNII).

- **Logistic Regression [147]**: a simple and efficient linear model for binary classification.
- **Decision Tree [204]**: a non-parametric supervised learning algorithm. It has a hierarchical tree structure, classifying samples following an ordering of attributes.
- **Random Forest [8]**: a bagging method that trains a multitude of decision trees and determines the final reason via majority voting.
- **XGBoost [4]**: a decision-tree-based ensemble machine learning algorithm that uses a gradient boosting framework.
- **DNN [89]**: a neural network with fully connected layers. We choose ReLU as the activation function and cross-entropy loss for model optimization.
- **CNN [113]**: a neural network that feeds the concatenation of the representation of categorical data (obtained via convolution operations) and numeric attributes to fully connected layers for predictions.
- **Wide & Deep Neural Network [38]**: a neural network that includes a deep module and a wide module. The wide module applies a linear model to improve the sparsity of categorical features and robustness of the model via cross-product feature transformations. For the deep module, a feed-forward neural network is applied for numeric features to improve the memory ability of the network.
- **DeepWalk [172]**: references a language model with the random walk algorithm for unsupervised node representation learning.
- **Node2vec [67]**: an unsupervised learning method, which applies a random walk for node representation.
- **GCN [107]**: a basic graph convolution neural network for node classification.
- **GraphSage [89]**: a general inductive framework that generates embedding by sampling and aggregating features from nodes' local neighborhoods.
- **GAT [89]**: applies the attention mechanism for feature aggregation.

- **SGC [254]**: a lightweight GCN, which removes nonlinear activation functions and collapses weight matrices between consecutive layers.
- **GCNII [23]** is an extension of the vanilla GCN, which applies initial residual and identity mapping techniques to alleviate the over-smoothing problem.

4.4.2.3 Implementation

For a fair comparison, we set the learning rate=0.001, regularizer=$1e-5$, batch size=500, embedding size $d = 8$, and select Adam as the optimizer for all deep learning models. Concretely, we set the number of graph convolution layers as 4. The size of the output of the graph convolution is kept the same as the input size, i.e.,$d \times m + n$. The size of FFN layers is $d \times m + n$ and 2. In addition, we set the inverse of regularization strength as $1e-4$ for logistic regression; the max depth is 5 for the decision tree; the number of trees for ensemble methods is 1000. For all the datasets, we extract 10% samples as a testing set, and the remaining as the training set.

4.4.3 Performance Comparison (RQ1)

Table 4.2: Experiment results on three datasets. We highlight the best performance and underline the sub-optimal results from the baselines for each comparison. OOM means Out of Memory.

Dataset	Home Credit			Lending Club			PPD		
	Recall	F1	AUC	Recall	F1	AUC	Recall	F1	AUC
LR	0.50163	0.48288	0.75361	0.54456	0.54161	0.69017	0.50000	0.48052	0.59130
DT	0.50075	0.48089	0.72718	0.53269	0.51864	0.69925	0.51874	<u>0.52024</u>	0.66042
RF	0.50115	0.48172	0.75074	0.52870	0.50943	0.70380	0.50000	0.48052	0.69286
XGB	<u>0.60400</u>	<u>0.55801</u>	0.77267	0.65707	0.58513	0.71616	**0.64745**	0.55218	0.69180
DNN	0.50000	0.47929	0.77183	0.50000	0.44317	0.70448	0.50000	0.48047	<u>0.71577</u>
CNN	0.50000	0.47922	<u>0.77457</u>	0.50000	0.44317	0.72961	0.50000	0.48050	0.71384
Wide & Deep	0.50000	0.47922	0.75824	0.50000	0.44317	0.70207	0.50000	0.48049	0.71109
DeepWalk	0.50475	0.49004	0.73620	0.50904	0.46885	0.61531	0.49820	0.47962	0.57000
Node2vec	0.50364	0.50370	0.61629	OOM	OOM	OOM	0.50168	0.48470	0.57227
GCN	0.50205	0.48348	0.74254	<u>0.88461</u>	0.87307	<u>0.96684</u>	0.50000	0.48051	0.66027
GraphSage	0.50084	0.48531	0.77232	0.73630	0.73955	0.90904	0.50000	0.48051	0.59451
GAT	0.50000	0.47920	0.76235	0.79781	0.83054	0.96094	0.50000	0.48051	0.61799
SGC	0.52008	0.51906	0.72078	0.87306	<u>0.87731</u>	0.89735	0.49982	0.48042	0.63664
GCNII	0.50000	0.47892	0.73059	0.86322	0.85776	0.96147	0.50000	0.48052	0.66953
MGCN	**0.61886**	**0.61811**	**0.78455**	**0.90588**	**0.87849**	**0.97546**	<u>0.63109</u>	**0.55987**	**0.71927**

Table 4.2 demonstrates the main results of all compared methods on three datasets. The major findings from the experimental results can be summarized as follows:

We can observe that our model MGCN outperforms all the baselines, especially on the Lending Club dataset (38% and 36% increases on recall and AUC, respectively, compared with XGBoost). Specifically, MGCN achieves a remarkable improvement (up to 15.28%) over the strongest baselines in terms of recall, which demonstrates the effectiveness of the multi-view loan application graphs and the data augmentation module we proposed.

Compared with other machine learning and deep learning methods, graph convolution-based models do not have a significant superiority except for the Lending Club dataset. Compared with other datasets, the number of attributes on the Lending Club is rather small, and the user's information is also limited. Hence, we believe information aggregation from neighbors will be more beneficial for the scenarios in which the raw information is limited. Additionally, DeepWalk and Node2vec are the worst-performing methods among all the models, which illustrates that the graph information obtained by the unsupervised learning method does not work well for node classification. Furthermore, deep learning methods do not achieve significant improvements on all datasets, while ensemble models, e.g., Random Forest and XGBoost, are still strong baselines and achieve comparable results.

For neural networks, Recall and F1 are virtually indistinguishable on all the datasets. Attribute to the imbalanced data distribution and missing values, it is difficult for all the baselines to achieve satisfactory results on all the datasets.

4.4.4 In-depth Analysis of MGCN (RQ2)

4.4.4.1 Impact of Embedding Size

As mentioned above, the categorical attributes should be projected into a high-dimensional latent space as an embedding first. Thus, we put the lens on the impact of different embedding sizes ($d = 5, 8, 16, 32$) on our model. As shown in Figure 4.4, when the embedding size is changed, there is no significant fluctuation for AUC on the three datasets, while Recall varies more. In general, when setting the embedding size to eight, our model achieves promising performance. We argue that a moderate embedding size could provide sufficient information capacity for the model while simultaneously avoiding the over-fitting problem.

4.4.4.2 Impact of Gate Mechanism and Graph Convolution Layers

In order to verify the effectiveness of the gate mechanism in MGCN, we set the number of graph convolution layers in the range of $\{1, 2, 3, 4\}$ to evaluate the model's performance. In

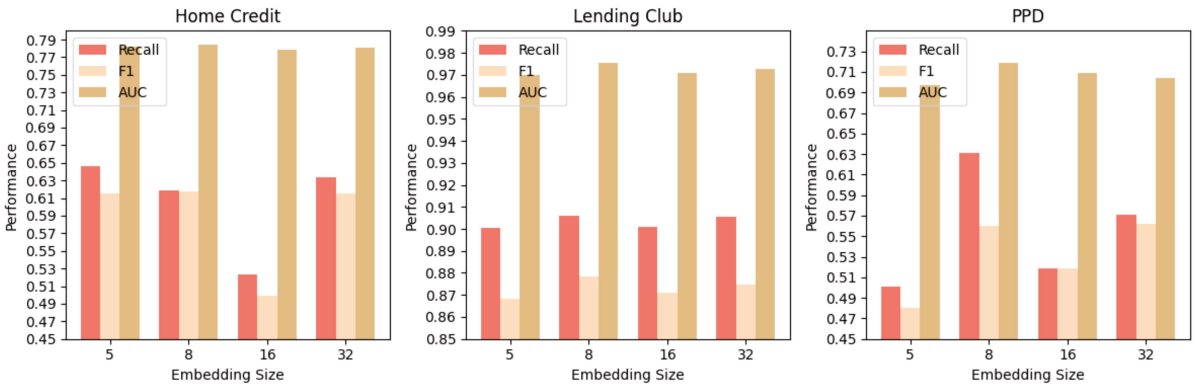


Figure 4.4: The effect of embedding size $\{5, 8, 16, 32\}$ of our model on three public datasets.

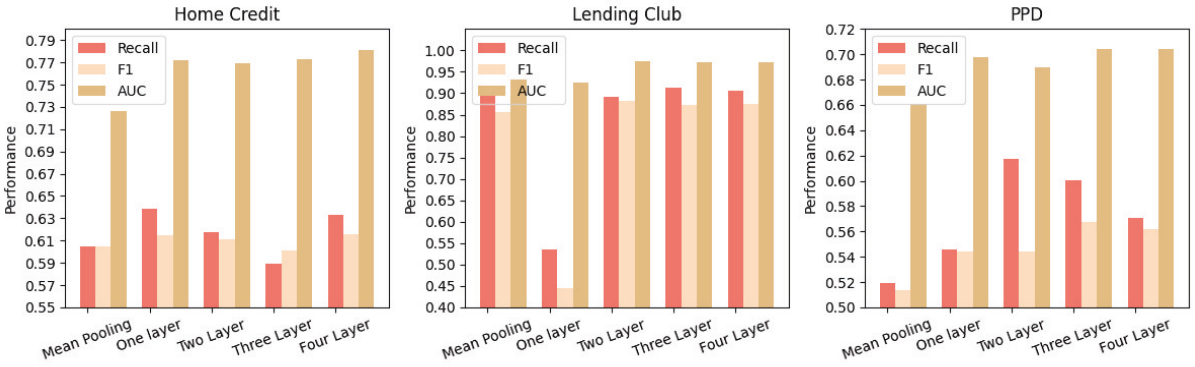


Figure 4.5: Performance of MGCN with different numbers of graph convolution layers and mean-pooling for feature aggregation.

addition, we replace the gate mechanism with mean pooling for the ablation experiment. The other hyperparameters and model structure remain the same as described above. We show the performance comparison results in Figure 4.5. It is obvious that the performance declines significantly when we replace the gate mechanism with mean pooling, indicating the usefulness of the gate mechanism for information aggregation. Besides, as we increase the layer of graph convolution, the AUC also rises, which demonstrates the effectiveness of high-level semantic information for loan default detection.

4.4.4.3 Impact of Shallow and Depth Features

As mentioned above, the deep layer contains rich semantic features and neighbor information. Hence, the feature representation from deep layers may facilitate the prediction of those samples whose neighbors are limited. In contrast, for the nodes that contain many neighbors, we assume the shallow layer's feature representation is sufficient for

classification. Consequently, we devise the gate mechanism for hierarchical feature aggregation and alleviate the over-smoothing problem simultaneously. In this experiment, we dive into analyzing the effectiveness of the gate mechanism. Specifically, we first remove the gate mechanisme and keep the other hyperparameters and model structure the same as described above. Then, applying one-hop layer feature representation (one graph convolution layer), multi-hop layer feature representation (four graph convolution layers), and mix-hops feature representation (four-hops feature representation for those nodes where the number of neighbors is less than 90% of the total nodes and one-hop feature representation for the remaining nodes) for loan default risk prediction. In addition, we also resort to one LAGs with the best result among all MLAGs to eliminate the effects of data augmentation for fair performance comparison. Results are shown in Table 4.3. Overall, our model achieves the best results on three datasets, which verifies the effectiveness of our proposed gate-based feature aggregation module. In addition, we can also observe that the mix-hop convolution achieves sub-optimal results on AUC, which confirms our aforementioned assumption that both deep and shallow features are equally important for default risk prediction. In Chapter 5.6, we will give an in-depth discussion regarding different fusion strategies in terms of effectiveness and efficiency.

Table 4.3: Impact of shallow layer, depth layer, and mix layer representation on loan default prediction. We highlight and underline the best and sub-optimal results.

Dataset	Metric	One-hop	Four-hops	Mix-hops	LAG
Home Credit	Recall	0.50021	<u>0.50205</u>	0.50000	**0.60345**
	F1	0.48143	<u>0.48348</u>	0.47920	**0.60887**
	AUC	0.75145	0.74254	<u>0.75691</u>	**0.76321**
Lending Club	Recall	**0.88627**	<u>0.88461</u>	0.87716	0.88192
	F1	0.85839	**0.87307**	<u>0.86405</u>	0.86210
	AUC	0.96447	0.96684	<u>0.96914</u>	**0.97117**
PPD	Recall	<u>0.50000</u>	<u>0.50000</u>	<u>0.50000</u>	**0.63109**
	F1	0.48051	<u>0.48052</u>	0.48051	**0.55987**
	AUC	0.68627	0.67561	<u>0.69666</u>	**0.70650**

4.4.4.4 Impact of Multi-view Loan Application Graphs

As mentioned in 4.3.1, the distribution between positive and negative samples is imbalanced. Thus, we construct multi-view loan application graphs for data augmentation. Specifically, by adjusting the similarity threshold between two nodes, we could change

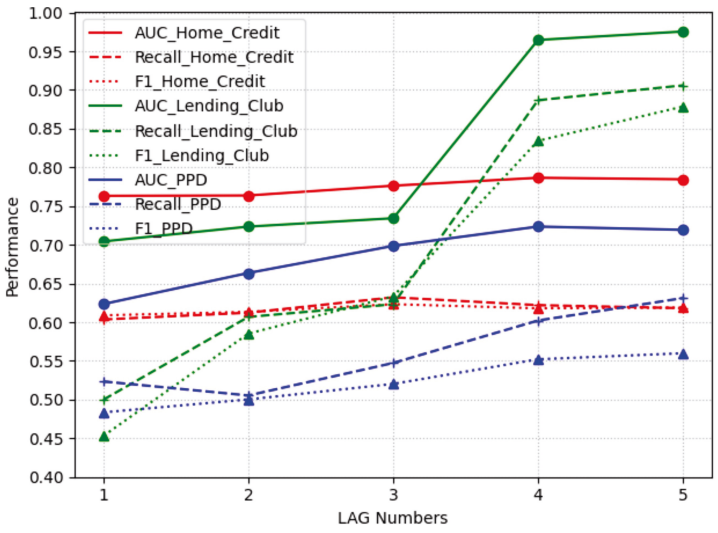


Figure 4.6: Performance of MGCN with different number of LAGs $1,2,3,4,5$ on three public dataset.

the graph structure and construct multiple loan application graphs, i.e., multi-view loan application graphs (MLAGs). Thus, the various node representations can also be yielded under different LAGs. Then, we collect the small sample representation from multi-view loan application graphs for model training. This process can be regarded as a small sample augmentation. In the experiment, we set the δ in the range of $\{0.05, 0.1, 0.15, 0.2, 0.25\}$. Hence, five LAGs are constructed, and we consider different numbers of LAGs simultaneously for small sample augmentation and loan default prediction. Note that when the number of LAGs is set to 1, our method becomes approximately equivalent to the proposed method in Chapter 3. As mentioned above, the other hyperparameters and model structures remain the same. The performance is presented in Figure 4.6. Overall, the performance of MGCN improved as the number of LAGs increased. Especially for the Lending Club dataset, when we increase the number of LAGs from one to five, the AUC rises from 0.70432 to 0.97546 (38.5% improvement). However, for the Home Credit Default Risk and PPD datasets, when the number of LAGs is over 3, the performance does not improve notably. We believe this is because the size of the Home Credit Risk dataset and PPD dataset is relatively smaller than Lending Club ($2,260,701$ vs $307,511$ for Home Credit Risk and $30,000$ for PDD, ref. Table 4.1 for details.). Consequently, a moderate graph number is sufficient to rebalance the data distribution. In general, we believe the data argumentation with multi-view graphs can mitigate the distribution imbalance issue to some extent, but we should carefully adjust the number of graphs to maximize performance improvement with limited computational costs.

4.4.5 The Effects of Sampling Strategies (RQ3)

As analyzed in Chapter 4.4.4.4, the defaulter records are small samples in a realistic scenario, and the data distribution is also imbalanced. Following previous works, we, therefore, investigated the impact of the sampling strategy on the performance of our model. To avoid the effects incurred by data augmentation of MLAGs, we only consider one LAG for model training. Upsampling, downsampling, and SMOTE strategies are devised on the training data. We define ξ as the ratio of the minority class to the majority class after resampling. Other hyperparameters and model structures remain the same as described above. As shown in Table 4.4, the sampling strategies do not contribute to performance improvement; in turn, SMOTE will lead to a decline in AUC. However, compared with Table 4.3, multi-view graphs data augmentation is more effective for performance improvement.

4.5 Conclusion

To summarize, we investigated the research on loan default risk prediction in a realistic scenario. By elaborately analyzing the characteristics of loan application records, we propose a graph convolution-based model MGCN, to solve the problem. First, considering numerous missing values in raw data, we attempt to construct loan application graphs. Consequently, similar loan application records could be aggregated as auxiliary information to improve the model's performance and robustness to missing values. Besides, to alleviate the effect of imbalanced distribution, multi-view loan application graphs are proposed for data augmentation. Furthermore, we devise a novel gate mechanism to fuse the deep and shallow features generated by the graph convolution and alleviate the over-smoothing issues. Extensive experiments are conducted on three real-world loan default prediction datasets with large numbers of feature fields. The results demonstrate the effectiveness of multi-view loan application graphs and the gate mechanism. Although we achieve promising results for default risk prediction in this work, we fuse the attribute embedding with a simple concatenation operation. In the following Chapters, we will discuss the application of graph neural networks in session-based recommendation. Further, we will explore the implicit relationships between any two attributes and fuse the attributes' embedding in a more sophisticated way.

Table 4.4: Performance of different sampling strategies on three datasets. We highlight the best performance for each comparison.

Dataset	Metric	Ratio	0.25	0.50	0.75	1.0
Home Credit	Recall	Upsample	0.59243	0.50000	0.69988	0.69991
		Downsample	0.50000	0.65075	0.69579	**0.70285**
		SMOTE	0.50000	0.51350	0.51731	0.50000
	F1	Upsample	0.60301	0.47919	0.56838	0.53515
		Downsample	0.47918	**0.61846**	0.56568	0.55149
		SMOTE	0.47919	0.50833	0.51517	0.47918
	AUC	Upsample	**0.77493**	0.76185	0.76959	0.76677
		Downsample	0.77315	0.77237	0.77090	0.77001
		SMOTE	0.76647	0.76454	0.76104	0.75882
Lending Club	Recall	Upsample	0.87896	0.89726	0.50000	0.84846
		Downsample	0.86974	**0.90453**	0.90392	0.86328
		SMOTE	0.88160	0.89483	0.86206	0.86465
	F1	Upsample	0.87215	0.86568	0.44467	0.82742
		Downsample	0.87435	**0.87443**	0.86412	0.82533
		SMOTE	0.87005	0.87554	0.85927	0.83774
	AUC	Upsample	0.97022	0.97083	0.94174	0.94174
		Downsample	0.97095	**0.97279**	0.96957	0.94516
		SMOTE	0.97053	0.97052	0.96381	0.94901
PDD	Recall	Upsample	0.50000	0.61156	**0.64837**	0.63997
		Downsample	0.50000	0.50000	0.50000	0.50000
		SMOTE	0.50000	0.51929	0.52793	0.50810
	F1	Upsample	0.48051	**0.57569**	0.49455	0.54587
		Downsample	0.48050	0.48048	0.48051	0.48051
		SMOTE	0.48050	0.51982	0.53223	0.50087
	AUC	Upsample	**0.71375**	0.71306	0.70454	0.70286
		Downsample	0.68783	0.69684	0.68317	0.65616
		SMOTE	0.70387	0.68739	0.52785	0.68045

CHAPTER 5

REVIEW OF SESSION-BASED RECOMMENDATION

5.1 Introduction[1]

With the prosperity and prevalence of the Internet, a surge of companies are allowed to provide their products or services via E-commerce, *e.g.*, Amazon, YouTube, Yelp, and LinkedIn. Although the vast array of information increases more probability for users to satisfy their personality requirements, it also exacerbates the issue of information overload, i.e., users often find it challenging to quickly identify their preferences and make decisions when facing an overwhelming influx of information. As an effective and efficient information filtering technology, Recommendation Systems (RSs) aim to mine users' *Point-of-Interest* (*POI*) based on their historical interaction records (*e.g.*, click, watch, read, add to cart, and purchase) and automatically recommend interested items. RSs have evolved into a prominent solution and attracted widespread attention from both academia and industry [61, 234, 256].

Considering that leveraging a user's historical information might be impractical in real-world scenarios due to (1) the user's preference will shift and evolve over time, and (2) it is difficult to realize a dynamic and efficient real-time recommendation.

However, most of the existing methods are committed to utilizing all historical information of users and capturing the long-term, statistical preference for recommendation [58]. It may not be practical in real-world scenarios due to (1) the user's preference

[1]This Chapter is based on our published work: Graph and sequential neural networks in session-based recommendation: A survey.

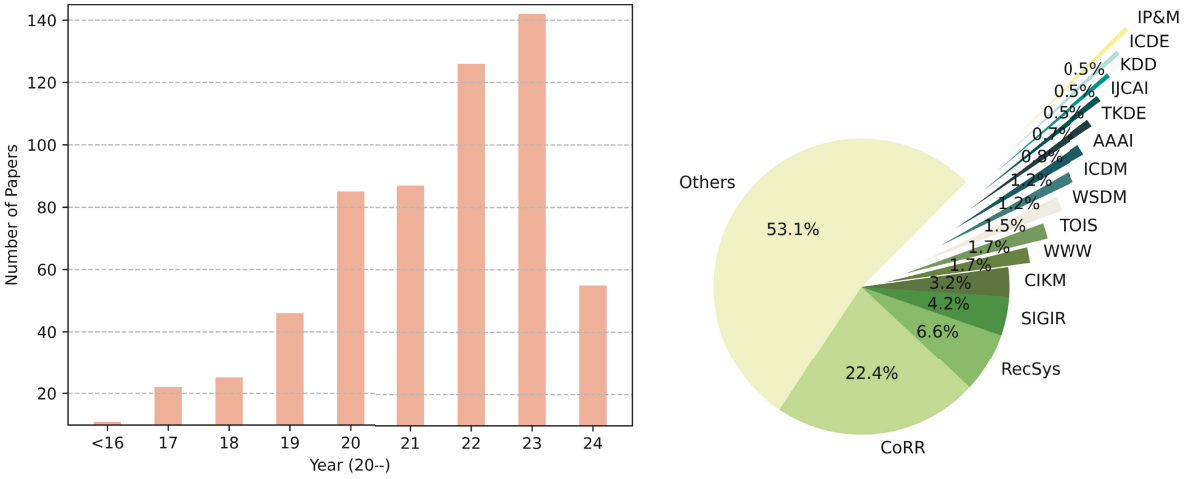


Figure 5.1: The statistics of publications with regard to SR. "<16" means 2016 and before. The bar chart (left) displays the number of published papers each year, and the pie chart (right) illustrates the percentage of papers published in each top venue.

will shift and evolve over time, and (2) those methods are difficult to model the user's ongoing behaviors and realize a real-time recommendation.

To bridge this gap, *Session-based Recommendation (SR)* [234] or *Session-aware Recommendation* [2] have emerged with increasing attention in recent years. As shown in Figure 5.2, Session-based recommendation aim to predict the next item a user is likely to engage with, leveraging the interaction sequence within the current session. By adjusting the recommendation results dynamically based on current interactions, SRmodels her short-term preferences to achieve an accurate and dynamic recommendation. Reviewing the development of SR, Modani *et al.* [158] first proposed the concept of *session* for a dynamic recommendation in 2002. On top of that, in 2005, Modani *et al.* [159] further devised a basic framework based on a bipartite graph, which is the pioneering work for the SR task. Since then, SR has gradually become a hot topic in recommendation research avenues. Figure 5.1 shows the number of papers (593 in total retrieved from DBLP Database[3] by June 2024) published in top venues. The amount of relevant work has increased significantly since 2019. In 2020 and 2021, the number of papers (85 and 87) was almost twice that of 2019 (46). In 2022 and 2023, the number rose to 126 and 142, respectively. Besides, RecSys, SIGIR, CIKM, WWW, WSDM, TOIS, and TKDE are the most popular target venues for SR.

[2]Quadrana *et al.* [179] emphasize that for *session-aware recommendation*, the user is anonymous *i.e.,* all historical interactions are known, but for *session-based recommendation*, we only focus on the current session. However, most studies do not distinguish these two terminologies.

[3]https://dblp.org/

Figure 5.2: The toy example of session-based recommendation. Session-based recommendation aims to predict the next item the user prefers to click, based on the interactions within the current session.

In general, early studies on SR specialized in popularity-based solutions [3] and machine learning-based methods, including KNN [51, 153], Markov Chain [188, 298] and matrix factorization [187, 198]. Benefit from the powerful ability of feature extraction and representation, deep learning solutions, such as sequential neural networks [68, 81, 103, 126, 185, 214] and graph neural networks (GNNs) [19, 22, 257, 260, 290, 309], become ubiquitous and achieve promising results for SR. More concretely, the sequential neural networks, *e.g.*, recurrent neural network (RNN), long short-term memory neural network (LSTM), and gated recurrent unit neural network (GRU), model the input session as a sequence and capture the order dependency among items for recommendation. In contrast, GNNs are required to predefine a graph based on sessions first, after that, items' correlations can be modeled via the information propagation and aggregation for recommendation.

Given the remarkable growth of SR research in graph and sequential neural networks, it is necessary and valuable to analyze the characteristics, categorize the existing efforts in a unified framework, formalize their focused problems, and summarize the general ideas, solutions, major challenges, and future directions thoroughly. Consequently, in this Chapter, we are devoted to proposing a systematic classification, conducting a detailed comparison of GNNs and sequential neural networks, and providing a comprehensive overview of their application in SR. The key contributions of this survey are summarized as follows:

- We standardize the concepts and definitions concerning SR and introduce representative graph structures commonly used in SR. Besides, we summarize the features of SR and compare the similarities and distinctions between SR and the sequential recommendation task.

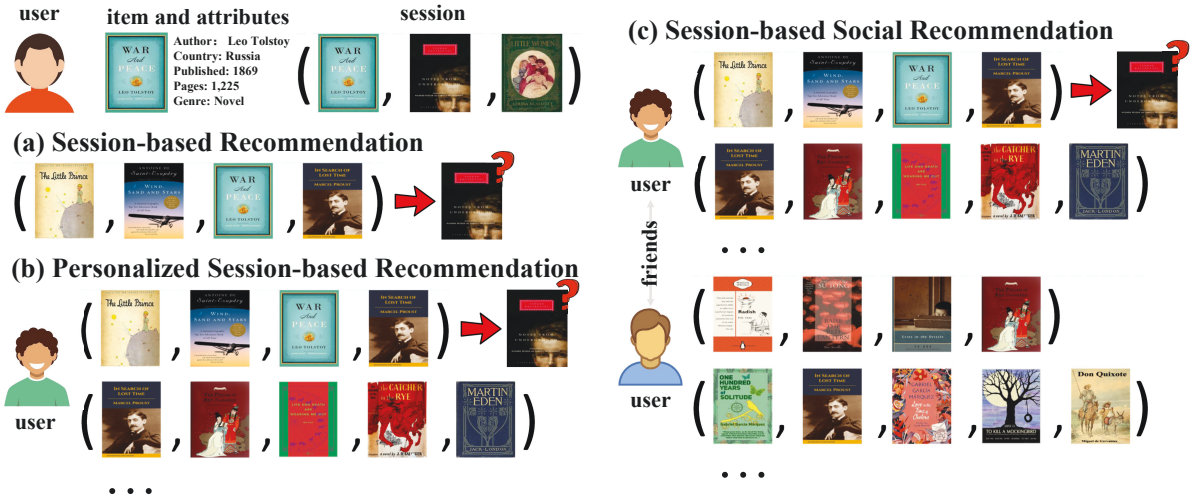


Figure 5.3: A toy example of different SR tasks.

- We propose a systematic category to organize the existing studies on sequential neural networks and GNNs for SR. A comprehensive analysis and comparisons with regard to the properties of these two mainstream methods are presented.
- Finally, we identify open challenges and discuss future directions on session-based recommendation.

5.2 Preliminaries

We will first clarify some basic definitions in SR, *e.g.*, intra-sessions, inter-sessions, and so forth. Various typical graph structures and the corresponding concepts are also briefly introduced.

5.2.1 Session Definitions

Based on the anonymity of a user's information, SR can be further divided into personalized session-based recommendation (PSR) and session-based social recommendation (SSR), which are illustrated in Figure 5.3. In addition, the definitions and the key concepts, *i.e.*, session, item, and attribute are listed below.

Item is the object that users interact with. It is usually formalized as an item ID in a recommendation system.

Attribute is the associated external information (or side information in [266]) of items or interactions. Specifically, the item-oriented attributes contain such as brands, categories,

text descriptions, and images. As for interaction-oriented attributes, like geographic information, interaction time and order, and behavior types (*e.g.*, search, click, add cart, buy, share, comment, etc) are most commonly used. These attributes could serve as auxiliary information for performance improvement.

Session refers to a user-interacted list consisting of items or services. The items in sessions are usually organized chronologically[5].

Session-based Recommendation (SR, depicted in Figure 5.3 (a)). Let $\mathscr{I} = \{i_1, i_2, ...i_n\}$ as the set of items, where n is the number of items. Each session $s = [i_1, i_2, ..., i_m]$ consists of a sequence of interactive items $i_k \in \mathscr{I}$ from a user. Thus, given a session s, the task of SR aims to generate probabilities (or scores) $\hat{\mathbf{y}}$ for all possible items. The items with a top-K recommendation score will be recommended.

Although collaborative filtering solutions, including item-based, user-based, and content-based methods, aim to predict users' next POI, these methods elaborate to capture the co-occurrence patterns based on the user-item interaction matrix, ignoring the sequential information of each click [212]. In contrast, session-based recommendation puts efforts into current session modeling, *i.e.,* only the ongoing session and the sequential patterns encapsulated in the sessions are considered. Thus, it will be more suitable for new users and timely, dynamic recommendations (more discussions can be found in Chapter 5.3.1).

Personalized Session-based Recommendation (PSR, illustrated in Figure 5.3 (b)). Let $\mathscr{U}$ be a set of users, for each user $u \in \mathscr{U}$, denote $\mathscr{S}^u = \{S_i^u\}_{i=1}^{n_u}$ as all the historical sessions of u, and n_u stands for the total number of sessions. Let $S_i^u = [i_{i,j}]_{j=1}^{m_i} \in \mathscr{S}^u$ as the i-th session of user u, and m_i stands for the total number of items in session S_i^u. We define S_c^u as the current session of user u, the previous sessions in the timeline are historical sessions denoted as $\mathscr{S}_h^u$. Thus, given all the historical sessions $\mathscr{S}_h^u$ of user u, the task of PSR is to predict the next interactive item of the current session S_c^u. In [70], the PSR is also named as streaming session-based recommendation.

Session-based Social Recommendation (SSR, illustrated in Figure 5.3 (c)). Based on PSR, we denote $\{u_k\}_{k=1}^{N(u)} \subseteq \mathscr{U}$ as the neighbors or friends of user u , $N(u)$ is the number of neighbors. Let the sessions of user u_k as $\mathscr{S}^{u_k}$. Thus, given the current session $\mathscr{S}^u$ of user u and all the sessions $\cup_{k=1}^{N(u)} \mathscr{S}^{u_k}$ from user u and his/her neighbors u_k, the task of SSR aims to predict the next interactive item of the current session $\mathscr{S}^u$.

[5] In most cases, this is true, while a few papers ignore the sequence information in sessions [91, 235]

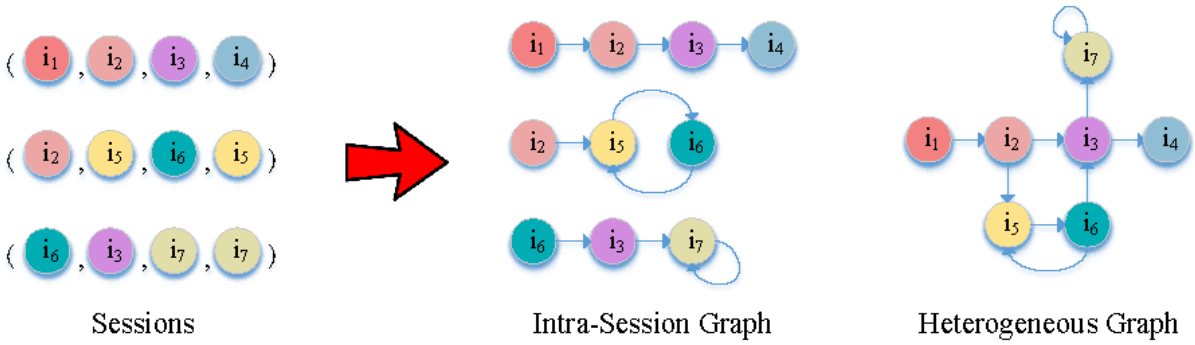


Figure 5.4: The diagram of the intra-session graph and the inter-session graph. We represent the items or nodes as solid circles and the edges as arrow lines.

5.2.2 Graphs Definitions

Graph-based SR endeavors to design dedicated graph structures to harness valuable information from neighbors for recommendation. We, thereby, introduce five typical graph structures in SR.

Digraph and Undigraph. Given a session s, each item in the session can be presented as a node. Besides, if a user clicks item i_j after item i_i in the session, we add a directed edge e_{ij} from node i to node j. Hence, we could organize those nodes and the directed edges between all adjacent items via a digraph. In contrast, if we add an undirected edge e_{ij} between node i and node j, we could construct an undigraph.

Intra-session graph. As shown in Figure 5.4 (left), given a series of sessions $\mathcal{S}$, we could construct intra-session graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ for each session where the node set is all unique items in $\mathcal{S}$. And $e_{ij} \in \mathcal{E}$ represents an edge where a user clicks item i_j after i_i in sessions.

Inter-session graph. As shown in Figure 5.4 (right), given a series of sessions $\mathcal{S}$, we could construct a unified inter-session graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ between all the adjacent items for all the sessions, where $\mathcal{V}$ and $\mathcal{E}$ indicate the set of nodes and edges, respectively. For an edge $e_{ij} \in \mathcal{E}$, it indicates the edge points from node v_i to node v_j.

Hypergraph. Given a series of sessions $\mathcal{S}$, we could construct a hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a node-set, which consists of all the unique items $\mathcal{I}$. We further define two adjacent metrics $\mathbf{A}^{n2e} \in \mathbb{R}^{M \times N}$ and $\mathbf{A}^{e2e} \in \mathbb{R}^{M \times M}$, where M and N are the number of hyperedges and items, respectively. Thus, if item i belongs to the hyperedge j, we set the element $a_{ij}^{n2e} = 1$, otherwise $a_{ij}^{n2e} = 0$. In addition, if hyperedge j and hyperedge k share the same items, we set the element $a_{jk}^{e2e} = 1$, otherwise $a_{jk}^{e2e} = 0$. Based on the hypergraph, the high-order information in sessions can be captured efficiently. As shown in Figure 5.5,

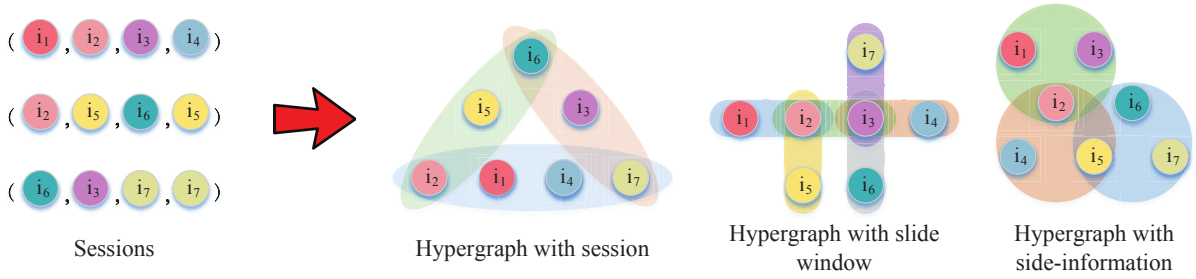


Figure 5.5: The diagram of hypergraphs. The shades of different colors represent different hyperedges.

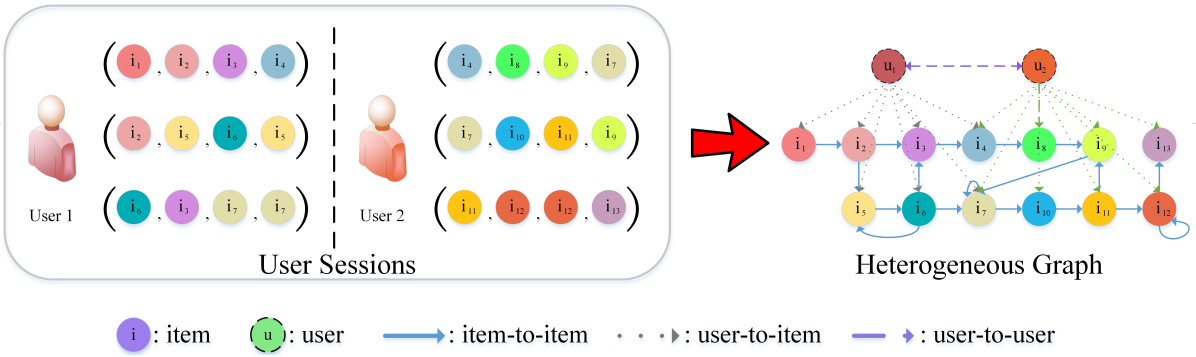


Figure 5.6: The user-item social session graph.

the hyperedge $e \in \mathcal{E}$ can be defined by: (1) the nodes sharing the same values of attributes (*i.e.*, hypergraph with attributes) [114]; (2) the items from the same session [261] (*i.e.*, hypergraph with session); (3) the item and its incoming items [132] (*i.e.*, hypergraph with incoming items); (4) belonging to a specific contextual window [229] (*i.e.*, hypergraph with slide windows); (5) a specific consecutive intent unit [69] (*i.e.*, hypergraph with intent unit). For instance, in [294], a hyperedge is defined based on the item side information, like item prices.

Heterogeneous graph. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, it consist of a node set $\mathcal{V}$ and an edge set $\mathcal{E}$. We define a node type mapping function: $\phi : \mathcal{V} \rightarrow \mathcal{M}_{node}$ and an edge type mapping function $\psi : \mathcal{E} \rightarrow \mathcal{M}_{edge}$. Denote $\mathcal{M}_{node}$ and $\mathcal{M}_{edge}$ are the sets of node types and edge types. Thus, if $|\mathcal{M}_{node}| + |\mathcal{M}_{edge}| > 2$, we define the graph $\mathcal{G}$ as a heterogeneous graph. Attributed to the various types of nodes and edges, the heterogeneous graph is capable of modeling more complicated structures than the homogeneous graph [241]. In SR, the user-item session graph [31, 170] and the item-attribute knowledge graph [157, 292] are the most commonly used heterogeneous graph structures, which can be elucidated below:

- **User-Item Social Graph.** Given a user set $\mathscr{U}$ and the historical interacted sessions of each user, we could construct a user-item session graph $\mathscr{G} = (\mathscr{I}, \mathscr{U}, \mathscr{E})$ based on user-item interacted records and the user's social network, where $\mathscr{I}$, $\mathscr{U}$ are item nodes and user nodes, respectively. In [170], it contains two types of edges, *i.e.*, *item-to-item* and *user-to-item*. Specifically, if item i_i and item i_j are neighbors in sessions, we add an edge between them. Thus, the item transaction relations can be captured via an item-to-item edge. Besides, if there are interactive records between user u and item i, we add an edge between them. Consequently, the user's historical interests can be modeled via user-item edges. Except for the above two types of edges in a user-item session graph, the social relations are also considered in [31, 231]. Specifically, if user u_i and u_j are friends or there exists some kind of interactive behavior (*i.e.*, follow, shared, etc) between them, a user-to-user edge can also be added in the graph $\mathscr{G}$, as shown in Figure 5.6. Moreover, if each session is regarded as a node, the user-to-session edge and item-to-session edge can also be created as an extension of the general user-item social graph [22].

- **Item-Attributes Knowledge Graph.** Given a session set $\mathscr{S}$, an item set $\mathscr{I}$, and its relevant attributes set $\mathscr{A}$, we could construct an item-attributed knowledge graph $\mathscr{G} = (\mathscr{I}, \mathscr{A}, \mathscr{E})$, which contains two types of nodes (item node and attribute node) and two types of edges (item-to-item and item-to-attribute). If there is an ordered tuple (i_i, i_j), we add an edge from item i_i to i_j. Furthermore, if item i_k contains the attribute a_i, we add an edge from i_k to a_i with the corresponding relation. As shown in Figure 5.7, we could construct a knowledge graph for a book recommendation scenario. The entities/nodes contain the title and the corresponding attributes, *e.g.*, author, country, and genre. Apart from that, we could also define two relations, *i.e.*, *published in* and *written by*, and add edges from items to attributes.

- **User-Behavior Session Graph**. Compared with the inter-session graph or the intra-session graph, the user-behavior session graph further defines the behavior relations, e.g., buy, click, between two adjacent items [200, 240].

- **Spatialtemporal Session Graph**. For some online service platforms, *e.g.*, Meituan and Yelp, the user's location and time information are significant for the precise recommendation. Consequently, the authors [131] propose a spatiotemporal graph containing item, session, location, and time, four types of nodes, for session-based recommendation.

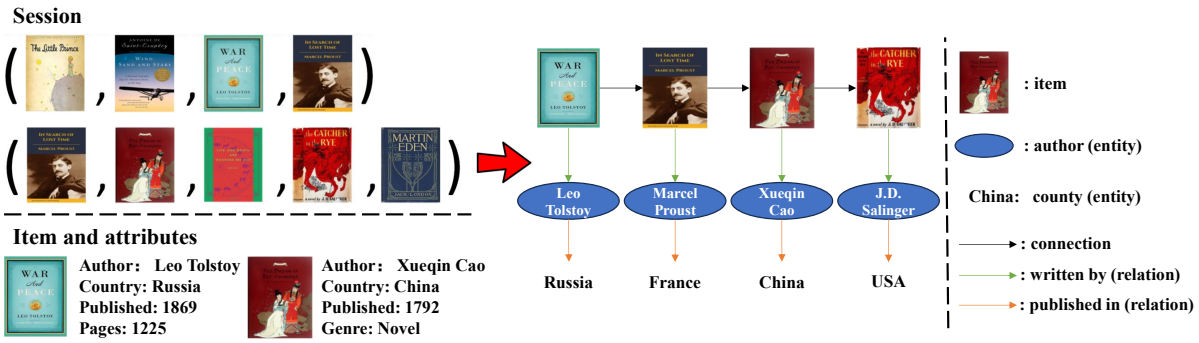


Figure 5.7: The item-attributes knowledge graph.

Overall, the connectivity and structure of the above graphs can be formalized as an adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ with $\mathbf{A}_{ij} \neq 0$ iff $(v_i, v_j) \in \mathscr{E}$ and $\mathbf{A}_{ij} = 0$ iff $(v_i, v_j) \notin \mathscr{E}$, where N is the total number of nodes. $\mathcal{N}_i$ indicates the neighbors of node v_i .

5.3 Features and Categorization of SR Approaches

We first summarize the features of SR. Then, a hierarchical categorization will be provided to organize different research lines.

5.3.1 The Features of SR

According to the aforementioned definitions, we summarize the features of SR as below.

- **Session Length.** Compared with the sequential recommendation, which organizes the user's all historical interaction records in chronological order, SR only concentrates on the current ongoing sessions, thus, the length of sessions (the number of contained items) is quite limited, *i.e.*, the median length of sessions is less than six for most popular public datasets. Therefore, Wang *et al.* [247] believe that sequential neural networks, *e.g.*, RNN, are unsuitable for SR.

- **Dynamic and Timely Recommendation.** Different from sequential recommendation, SR specializes in the current session and focuses more on users' short-term interest modeling instead of the interest evolution and long-term dependencies between items in the sequential recommendation. Therefore, SR aims to make dynamic and timely recommendations for the ongoing session. Graph neural networks are essential for SR, whereas large-scale graphs with billions of nodes and edges require construction in real-world applications. Since each node carries diverse

features, direct prediction on such massive graphs are nontrivial, making scalable graph design crucial for SR.

- **Adjacent Dependency.** Although the items in sessions are organized chronologically, there are no obvious order patterns [91]. Consequently, many studies [69, 93, 261, 305] apply GNNs to model the co-occurrence between two items, i.e., the item transition patterns in sessions are rooted in the co-occurrence within items, instead of reflecting by sequential pattern.

- **Anonymous.** As we discussed above, SR pays more attention to the current session modeling, thus, the user information and historical interactions are unavailability. Nevertheless, some papers [31, 170, 177, 207] propose personalized session-based recommendation (PSR) or session-based social recommendation (SSR), which consider the user's historical records are non-anonymous and use whole historical sessions as auxiliary information to improve the performance of current session recommendation, these approaches have not yet become mainstream in SR.

Overall, the length of sessions is rather limited against sequential recommendation, and there is no obvious dependency between two items in a session, despite the items being organized chronologically. Hence, most existing studies focus on item correlation modeling with GNNs for a dynamic and timely recommendation. Additionally, although some efforts apply users' social networks or historical records as auxiliary information for SR, modeling the current session is still the mainstream of this research venue. Moreover, more and more studies attempt to use external information for recommendation. Hence, a surge of sophisticated model architectures emerges for side-information fusion.

5.3.2 Classification of SR Methods

The taxonomy of existing SR methods is presented in Figure 5.8. In this Chapter, we summarize the representative solutions as three main categories: *i.e.*, *conventional methods*, *sequence-based methods*, and *graph-based methods*. The conventional methods can be further divided into *popularity-based methods*, *e.g.,* KNN, and matrix factorization. To be specific, the idea of *popularity-based methods*, *e.g.*, POP, S-POP, is to recommend the popular items to users while ignoring the cold start items. As the extension of popularity-based methods, frequent pattern or association rule mining approaches, *e.g.*, FP-tree, are applied, which mine the frequent items and patterns from raw data for recommendation [199, 276]. As for KNN-based methods [51, 153], they rely on similarity

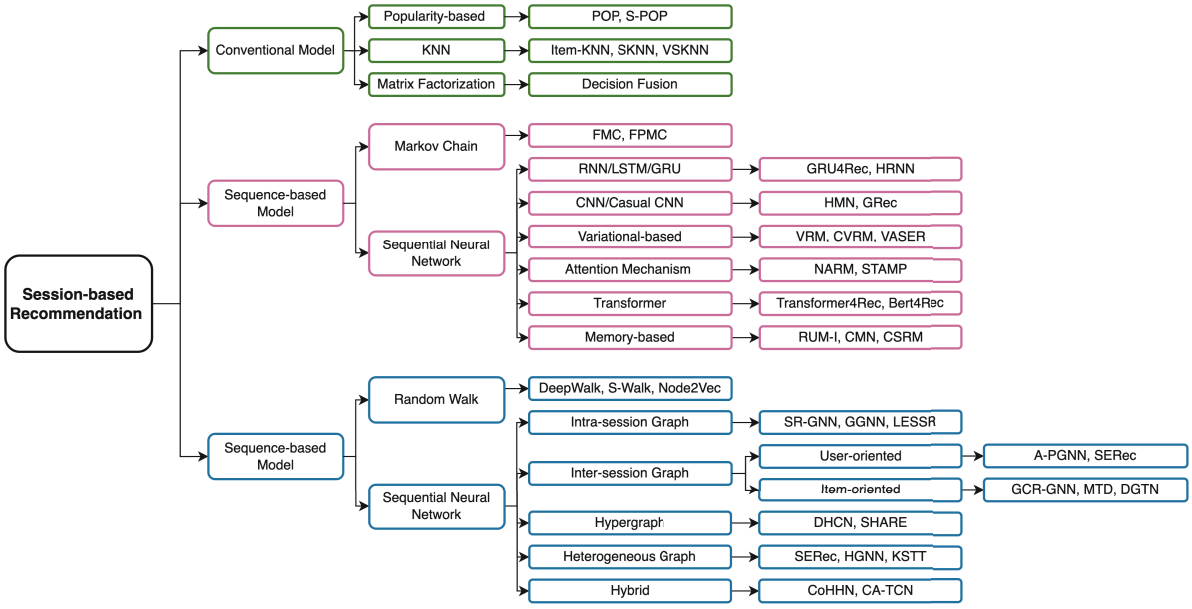


Figure 5.8: The categorization of SR approaches. The gray boxes are representative models for each class.

calculation for recommendation, which can also be divided into item-oriented KNN and session-oriented KNN. The item-oriented KNN measures the similarity between a target item and candidates and recommends the closest (*i.e.*, most similar) items to a user. The session-oriented KNN first selects the most similar sessions via similarity calculation. Thus, the items from those sessions are collected as candidates for further recommendation. Matrix factorization is also a mature method for SR, in which a user's and items' latent feature representation are learned from the user-item interaction matrix for recommendation [139, 187].

Apart from that, the Markov Chain is a prominent recipe in the early stage, which recognizes the next interaction prediction as a Markov Decision Process (MDP) and learns a state transfer matrix for recommendation [188, 198, 298]. For instance, FMC [198] is a pioneering work that extracts sequential patterns to predict the next item based on Markov models. FPMC [188] models sequential behavior between every two adjacent items via the personalized probability transition matrix factorization. Although the Markov chain achieved remarkable success for SR in the early years, the strong assumption that the next clicked item only depends on the previous one confines the development of this method.

Attributed to the powerful ability in feature representation, sequential neural networks, *e.g.*, RNN, LSTM, and GRU, are carried out consecutively for SR [81, 179, 217, 232].

For instance, GRU4REC [81] is the first that apply RNNs to model the session information for the next item recommendation, which stacks multiple GRU layers and applies a session-parallel mini-batch training strategy for performance improvements. HRNN [179] develops hierarchical RNNs with inter-session information to achieve personalized session-based recommendations. As a follow-up study [217], Tan *et al.* enhanced RNNs by leveraging data augmentation. Hidasi and Karatzoglou [80] proposed a new class of loss functions combined with modified sampling strategies to improve the performance of SR. Guo *et al.* [68] proposed a Hierarchical Leaping Network (HLN) with Leap Recurrent Unit (LRU) to decide whether the current item should be skipped or not. Being endowed with the property of global information modeling, the attention mechanism is also applied for SR. NARM [126] applies a hybrid encoder with an attention mechanism to model the user's sequential behavior and capture the main intentions in the current session. Liu *et al.* [149] took a similar idea but replaced the recurrent neural network with a multi-layer perception (MLP) and proposed STAMP to enhance the influence of the latest interests in sessions for both long-term and short-term interests capture. Furthermore, CNN and Causal CNN are proposed to capture n-gram multi-scale features for item representation and recommendation [205, 285]. Attribute to the remarkable performance achieved by Transformer [224], some researchers elaborated on Transformer for SR, *e.g.*, Transformer4Rec [52] and BERT4Rec [214]. Other studies, like memory neural networks [56, 232] and Variational Encoder (VAE), are also explored. For instance, in [245, 308], the latent variable module is introduced into sequential neural networks to improve the variation of recommendation. To sum up, we categorize the existing sequential neural networks with six aspects, as shown in Table 5.1.

Although sequential methods achieve satisfactory performance in SR, they are committed to the session's sequential information learning while ignoring the implicit dependency between items. Consequently, graph-based methods, a more flexible solution for transition pattern modeling, are accommodated for SR. Owing to the difference in item representation learning, graph-based methods can further be divided into random walks and GNNs. For random walk [67, 172], in general, given a series of sessions, an item-item adjacent graph is constructed first. Then, different random walk strategies are adopted with unsupervised learning to generate item representations. For instance, DeepWalk [172] empirically learns a low-rank transformation of a normalized Laplacian matrix for recommendation. Node2vec [67] learns node representations based on the word2vec model for recommendation. To balance both the accuracy and scalability of SR, S-Walk [41] proposes a random walk with a restart strategy to capture inter-session

Table 5.1: Comparison of representative sequential neural networks concerning six aspects, including motivation, session, sequential modeling, prediction, loss function, and datasets.

Motivation	Session	Sequential Modeling	Prediction	Loss	Datasets	Paper
Repeat Item	Current	GRU, Att	Inner production+softmax	CE	Dig, Yoo, FM	[185]
RNN for SR	Current	GRU	MLP	TOP1	Yoo, VIDEO	[81]
BERT for SR	Current	Att	Inner production+softmax	CE	Mov, Steam, Amz	[214]
Vanilla Attention for SR	Current	Att, GRU	Inner production+softmax	CE	Retail, Yoo	[126]
	Current	Att	Inner production+softmax	CE	Mov, Amz	[103]
Multi Interests	Current	Variant GRU	Inner production+softmax	CE	Yoo, FM	[68]
	Current	CNN, Att	Inner production+softmax	CE	Dig, Yoo	[205]
Long-term and Short-term	User	GRU, Att, Gate	Inner production+softmax	CE	Tmall, Oth	[32]
	Current	Att	Inner production+softmax	CE	Yoo, Oth	[52]
	Current	Att	Inner production+softmax	CE	Dig, Retail	[287]
	Current	Att	Inner production+softmax	CE	Dig, Yoo	[149]
Loss Function	Current	GRU	MLP	Ranking-max	Yoo, VIDEO, Oth	[80]
	Current	Att	Cosine+softmax	RDM	Dig, Yoo, Tmall, Now	[86]
	Current	MLP	Inner production+softmax	List-wise Ranking	Dig	[253]
	Current	Att	Inner production+softmax	List-wise Ranking	Dig, Yoo, Oth	[251]
	Current	Att	Inner production+softmax	Adaptive weight CE	Tmall, Retail	[165]
Neighbor Sessions	User	GRU	MLP	TOP1	Xing, VIDEO	[179]
	User	MLP	Inner production+softmax	CE	Tmall	[91]
	User	GRU, Att, MF	Inner production+softmax	CE	Gow, FM	[70]
	Sharing	GRU, KNN	MLP	CE	Yoo	[99]
	Time Close	GRU, Att, Gate	Inner production+softmax	CE	Yoo, FM	[232]
	Sim (Jaccard)	Att	Inner production+softmax	CE	Retail, Yoo	[154]
	Sim (Cosine)	GRU	Inner production+softmax	CE	Dig, Yoo	[169]
	Sim (Inner Product)	RNN, Att	Inner production+softmax	CE+BPR	Dig, Yoo	[248]
External Info (Position)	Current	Att	Inner production+softmax	CE	Mov, Steam	[195]
External Info (Attribute)	Current	Att	Inner production+softmax	CE	Amz, Oth	[266]
External Info (Attribute)	Current	Att	Inner production+softmax	CE	Dig, Oth	[197]
External Info (Attribute)	Current	Att	Inner production+softmax	CE	Amz, Oth	[97]
External Info (Future Interaction)	Current	CausalCNN	Inner production+softmax	CE	Mov, Oth	[285]
External Info (Key Words)	Current	GRU	Inner production+softmax	CE	Oth	[151]
External Info (Description)	Current	GRU	MLP	CE	Mov	[174]
External Info (Social Relations)	Current	Att	Inner production+softmax	CE	Gow, Oth	[166]

Illustration of abbreviations:
(1) Motivation:
a) Repeat Item: There is a certain probability that the target item will appear in the current session. Hence, a dedicated module is adopted for repeat item prediction. **b)** RNN/Attention/BERT for SR: The pioneering works that apply these models for SR. **c)** External Info: External information (*e.g.*, position, attributes of items, interaction information, keywords of items, text description, users' social information) is introduced as auxiliary signals for SR. **d)** Multi Interests: As the user's interests are dynamic and diverse, a single interest representation vector is insufficient for SR. Therefore, multi-interest representations are proposed. **e)** Long-term and Short-term: Users' long-term and short-term interests should be captured simultaneously for fine-grained intention recognition. **f)** Loss Function: improving loss functions for SR. **g)** Neighbor Sessions: Similar sessions are introduced for SR argumentation.
(2) Neighbor Session:
a) Sharing: the sessions sharing the same items with the current session are considered as neighbor sessions. **b)** User: a user's all historical sessions are selected as neighbors sessions. **c)** Current: only consider the current session for SR. **d)** Sim: calculate the similarity between sessions for neighbor session selection. **e)** Time Close: the sessions closed in the timeline are selected as neighbor sessions.
(3) Sequential Modeling:
a) Att: variant attention mechanisms. **b)** Gate: Gate mechanism. **c)** MF: Matrix factorization.
(4)Datasets:
a) Dig: Diginetica. **b)** Yoo: Yoochoose (Yoochoose 1/4, Yoochoose 1/64 are the most common for SR). **c)** Gow: Gowalla. **d)** FM: Last.FM. **e)** Retail: Retailrocket. **f)** Now: Nowplaying. **g)** Amz: Amazon. **h)** Mov: MovieLens. **i)** Oth: some other less common datasets in SR, *e.g.*, Tianchi, G1 news, CLASS, Delicious, OTTO.

and intra-session relations. In contrast, attributed to the information propagation and aggregation, GNNs can capture multi-hop contextual information between items for representation learning. Given that, GNNs demonstrate great superiority against sequential neural networks. As introduced in **Chapter** 5.2.2, based on the variation of graph structures, they can be divided into intra-session graphs, inter-session graphs, hypergraphs, heterogeneous graphs, and hybrids. A carefully selected representative GNN-based method for SR is presented in Table 5.2. In summary, GNNs offer more expressive representations than sequential models for session data. Sequential neural networks tend to better capture temporal order but may struggle with complex item-item dependencies. GNNs are effective in capturing item co-occurrence and relational struc-

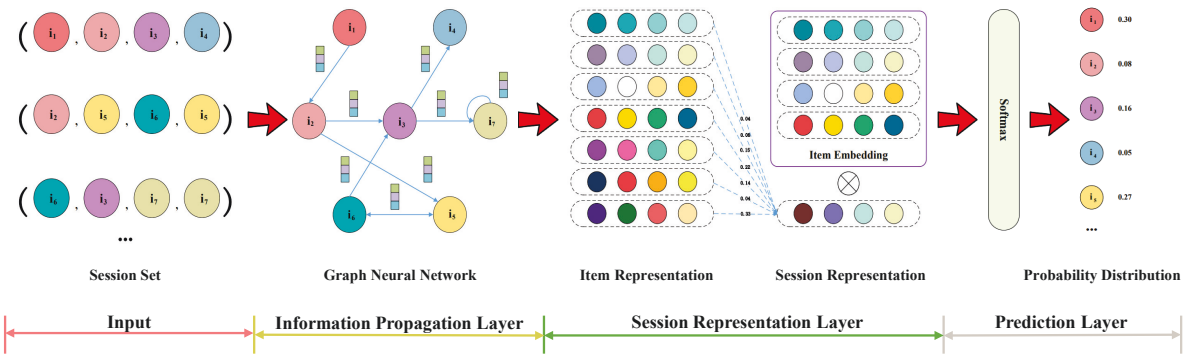


Figure 5.9: The framework of GNNs for SR.

tures but may lose fine-grained sequential patterns. In the following, we will present a comprehensive framework of GNN-based methods in SR.

5.4 The Framework of GNNs for SR

Encouraged by the powerful ability of GNNs in complicated graph data modeling across a wide spectrum of applications (*e.g.*, credit default risk prediction [137], traffic prediction [102], drug discovery [122], and multivariate time series imputation [36]), in 2019, Wu *et al.* [257] proposed SR-GNN, a pioneering work applying GNN for SR. Since then, a plethora of GNN-based studies have emerged. In general, the whole process of GNNs for SR includes five key modules: session selection, graph construction, information propagation and aggregation, session representation, and target item prediction, shown in Figure 5.9.

Specifically, given a session set $\mathcal{S}$ or a session s, a graph $\mathcal{G}$ should be constructed first. Then, variants of GNNs are designed to obtain the item representation via information propagation and aggregation. The item representation $\mathbf{H}$ will be fed into the session representation layer $\mathrm{SR}(\cdot)$ for session representation generation. Finally, the predicted item probability $\hat{y}$ is calculated via inner product for recommendation. This process can be formalized as below.

$$\begin{aligned} \mathbf{H} &= \mathrm{GNN}(\mathbf{X}) \\ \mathbf{S} &= \mathrm{SR}(\mathbf{H}) \\ \hat{y} &= \mathrm{P}(\mathbf{S}, \mathbf{X}) \end{aligned} \tag{5.1}$$

Table 5.2: Summarization of representative GNN-based studies with motivation, session, graph construction, item representation, session representation, loss function, and datasets.

Motivation	Session	Graph	Item Rep.	Session Rep.	Loss	Datasets	Paper
GNN for SR	Sharing	Intra	GCN, GRU	Att	CE	Dig, Yoo	[257]
Historical Info	User	Intra	GCN,GRU, Att	Att	CE	Xing, Reddit	[290]
	User	Inter, Intra	GCN, GGNN	Att	CE	Dig, Retail, Oth	[309]
User and Social Info	User	Het-Social	BiLSTM, GAT	Att	CE	Dig, Tmall	[22]
	User	Het-Social	Att, AvgP, Gate	Att	CE	Gow, Oth	[31]
	User	Het-Social	GAT	Att, Gate	CE	FM, Xing, Reddit	[170]
	User&Friends	Het-Social	LSTM, RNN, Att	Att, AvgP, Gate	CE	Oth	[207]
	User&Friends	Het-Social	GCN, GRU, Att	Att,	CE, Oth	Gow, Oth	[231]
High-order Connection	Sim (Duplicate)	Inter, Intra	GCN	Att	CE	Dig, Yoo	[305]
	Sim (Cosine)	Inter, Hyper	GCN, GRU	Att	CE	Dig, Yoo, Tmall	[100]
	Sim (Last item)	Inter, Intra	GCN, GRU	Att	CE	Dig, Yoo	[24]
	Current	Hyper	Att	Att	CE	Dig, Yoo, FM	[69, 229]
	Current	Variant Intra	GCN, GRU, Att, Gate	Att	CE	Dig, Yoo	[168]
	Current	Variant Intra	GCN, GRU, Att	Att	CE	Dig, Yoo, FM	[30]
	Current	Variant Intra	GCN	Att	CE	Dig, Gow, FM	[275]
	Current	Hyper	GAT	Att	CE, Oth	Tmall, FM	[132]
	Current	Hyper	GAT	Att	CE, Oth	Tmall, FM	[132]
	Sharing	Inter	GCN	AvgP	CE	Dig, Yoo, Retail	[93]
	Sharing	Inter, Session	GCN, CausalCNN	Att, GAT	CE	Dig, Yoo	[277]
	Sharing	Hyper, Session	GCN	Att, AvgP	CE, InfoNCE	Dig, Tmall, Now	[261]
	Sharing	Inter	GCN	Att	CE, InfoNCE	Tmall, Retail, Dig	[260]
	Sharing	ϵ-Neighbor	Att	GRU	CE	Dig, Yoo, Retail	[301]
	Sharing	Intra, ϵ-Neighbor	Routing, RW	GRU	CE	Dig, Yoo	[300]
	Sharing	Intra, ϵ-Neighbor	GAT	Att	CE	Dig, Tmall, Now	[246]
	Sharing	Intra, Inter	Sparse Att	Sparse Att	CE	Dig, Tmall, Now, Yoo	[175]
	Sharing	Intra, Inter	Att, GRU	Att	CE, InfoNCE, Oth	Dig, Tmall, Now	[211]
External Info (Behavior)	Current	Het-Behavior	Att	Att	CE	Dig, Gow, FM	[312]
	Current	Het-Behavior	AvgP	AvgP	CE	Yoo, Oth	[240]
	Current	Het-Behavior	GCN, GRU, Att	Gate	CE	Oth	[286]
	Current	Hyper	GAT, GRU, Att	Att	CE	Tmall, Yoo, FM	[200]
External Info (KG)	Current	Het-KG	GAT, TransR	Att	CE, BPR	Dig, Yoo	[292]
	All	Het-KG, Intra, Inter	GAT	Att	CE	Amz, Yelp, Oth	[26]
External Info (KG, Behavior)	Current	Het-KG	GCN, GRU, TransH	Att	CE, BPR	Jdata	[157]
External Info (Att)	Current	Variant Intra, Hyper	GAT	Att	CE	Dig, Tmall, Oth	[114]
	Current	Het-Behavior,Het-Attribute	GAT	Att	CE	Amz, Oth	[27]
External Info (Time, Location)	All	Het-Spatialtemporal	GAT, Att	Att	CE	Oth	[131]
External Info (Order)	Current	Intra	GAT, GRU	Att	CE	Dig, Yoo	[176]
External Info (Price)	Sharing	Hyper, Het-Attribute	Att, Gate	Att, Gate	CE	Dig, Oth	[292]
External Info (Other domain)	Sharing	Intra, ϵ-Neighbor	GAT, GRU	Att	CE	Dig, Yoo, Gow, FM	[19]
Multi Interests	Current	Intra	GGNN, Att, Gate	Att	CE, Oth	Retail, Yoo, Jdata	[201]
	Current	Intra	GCN, GRU	Att	CE, Oth	Dig, Yoo, Now	[118]
	Current	Intra	GCN, GRU	Att	CE	Dig, Yoo	[284]
Efficiency	Sharing	Inter, Variant Intra	RW	RW	CE, Oth	Dig, Retail, Oth	[41]
	Sharing	Inter, Intra	GCN, GRU, RW	Att	CE	Dig, Yoo	[55]
	Sharing	Variant Intra	GCN	Att	CE, InfoNCE	Tmall, FM, Retail	[171]
	User	Intra	GCN	Att	CE	FM, Gow	[177]
	Current	Intra	Att, GRU	Att	CE	FM, Gow, Dig	[291]

Illustration of abbreviations:
(1) Motivation:
a) GNN for SR: A groundbreaking work that applies GNN for SR. **b)** Historical Info, User and Social Info: Introducing neighbor sessions via different strategies *e.g.*, social network, for SR. **c)** External Info: Introducing external information *e.g.*, interaction behavior, item's attributes, time and location, knowledge graph, other domain data, and order information for SR. **d)** Multi Interests: Construct multi-interest representations for each user for SR. **e)** High-order Connection: One-hop connection is insufficient. Consequently, sophisticated graph structures (*e.g.*, self-loop, shortcuts) are designed for global information or multi-hop information modeling. **f)** Efficiency: Focusing on low computational complexity and timely recommendation.
(2) Session Selection:
a) Sharing: the neighbor sessions that contain the same item as the current session. **b)** User: a user's all historical sessions are selected as neighbor sessions. **c)** Current: only consider the current session for SR. **d)** Sim: calculate the similarity between two sessions or observe the last item in sessions for neighbor session selection. **e)** All: defines all the training sessions as neighbors.
(3) Graph Construction:
a) Intra: Intra-session graph. **b)** Inter: Inter-session graph. **c)** Het-Social: Heterogeneous graph with social relations. **d)** Het-Behavior: Heterogeneous graph with different interaction behavior types. **e)** Het-KG: Heterogeneous graph with knowledge graph. **f)** Het-Attribute: Heterogeneous graph with item correspondent attributes. **g)** Het-Spatialtemporal: Heterogeneous graph with time and location. **h)** Hyper: Hypergraph. **i)** Session: Introduce the session as a node in graph construction. **j)** ϵ-Neighbor: construct a graph based on the ϵ-Neighbor strategy. **k)** Variant Intra: intra-session graph with a virtual star or self-loop edge and shortcut edge.
(4) Information Propagation and Session Representation:
a) Att: various attention mechanisms. **b)** AvgP: Average pooling. **c)** Gate: Gate mechanism. **d)** RW: Random walk.
(5) Loss Function:
a) CE: Cross-entropy loss. **b)** Oth: auxiliary loss functions for multi-task learning, e.g, link prediction, matrix learning, ELBO loss for item representation distribution reconstruction, loss function with regularization term, etc.
(6)Datasets:
a) Dig: Diginetica. **b)** Yoo: Yoochoose (Yoochoose 1/4, Yoochoose 1/64 are the most common for SR). **c)** Gow: Gowalla. **d)** FM: Last.FM. **e)** Retail: Retailrocket.
f) Now: Nowplaying. **j)** Amz: Amazon. **h)** Oth: other datasets in SR, *e.g.*, Wechat, Cosmetics, Aotm, 30music, JD, Trivago, Delicious, Foursquare.

Table 5.3: The statistical characteristics of commonly used public datasets after preprocessing for SR. # means the total numbers, Avg. calculates the mean value.

Domain	Dataset	# sessions	# interactions	# items	Avg. session length
E-commerce	Yoochoose[7]	1,375,128	5,426,961	28,582	3.95
	Tmall[8]	1,774,729	13,418,695	425,348	7.56
	Diginetica[9]	780,328	982,961	43,097	5.12
	RetailRocket[10]	59,962	212,182	31,968	3.54
Music	Last.FM[11]	169,576	2,887,349	449,037	17.03
	NowPlaying[12]	27,005	271,177	75,169	10.04
Job Position	Xing[13]	91,683	546,862	59,121	5.78
Check-in	Gowalla[14]	830,893	245,157	6,871	4.32

5.5 Datasets, Evaluation Metrics

We first analyze the statistical characteristics of publicly available real-world datasets. Besides, the evaluation metrics concerning accuracy and diversity are also presented.

5.5.1 Public Datasets

According to the existing studies on SR, we summarize eight popular public datasets that cover E-commerce, Music and Video, Job Position, and Check-in scenarios. The statistical characteristics after preprocessing[6] are presented in Table 5.3.

- **Yoochoose** is the dataset of RecSys Challenge 2015, which contains a stream of user clicks and buy events on an online webshop within six months. Since the set of Yoochoose is extremely large, the most recent portions 1/64 and 1/4 subsamples of all sessions are usually used as the training set, denoted as "Yoochoose1/64" and "Yoochoose1/4", respectively [19, 30, 149, 185, 240, 257].

- **Tmall** comes from the IJCAI-15 competition, which contains users,Äô shopping logs on the Tmall online shopping platform.

[6]Following existing studies, we filter out all sessions whose length is 1 and items appearing less than 5 times.

[7]https://www.kaggle.com/chadgostopp/recsys-challenge-2015

[8]https://tianchi.aliyun.com/dataset/dataDetail?dataId=42

[9]https://competitions.codalab.org/competitions/11161

[10]https://www.kaggle.com/retailrocket/ecommerce-dataset

[11]http://millionsongdataset.com/lastfm/

[12]https://www.kaggle.com/chelseapower/nowplayingrs

[13]http://2016.recsyschallenge.com/

[14]http://snap.stanford.edu/data/loc-gowalla.html

- **Digineitca** is a personalized e-commerce research challenge dataset released in CIKM CUP 2016. The dataset contains transition histories, which are suitable for session-based recommendation.
- **RetailRocket** contains user behavior data and item properties collected from a real-world e-commerce website.
- **Last.FM** is a music artist recommendation dataset published by Celma *et al* [79].
- **NowPlaying** dataset comes from [288] and is created from music-related tweets, which illustrate the music-listening behavior of users.
- **Xing** Recsys Challenge 2016 Dataset [15] contains user interactions (click, bookmark, reply, and delete) on a job posting platform for 770k users over an 80-day period.
- **Gowalla** encompasses the check-in data and social network information from a location-based social networking website. It is widely used for point-of-interest recommendations.

Observing Table 5.3, we could find that the length of sessions in existing public SR datasets is rather limited. As exemplified by the average/median lengths of sessions, they are 5.12/4.0 and 3.95/3.0 for Diginetica and Yoochoose, the two most popular session-based datasets. We assert that there is no clear order dependency between any two items, suggesting that modeling the items' correlation with GNNs rather than sequential information with sequential neural networks may be more appropriate for SR [29].

5.5.2 Evaluation Metrics

Evaluation Metrics for Accuracy. Accuracy aims to measure the alignment of recommendation results and user interests. In general, HR@K (Hit Rate calculated over top-K items), MRR@N (Mean Reciprocal Rank calculated over top-K items), and NDCG@K (Normalized Discounted Cumulative Gain calculated over top-K items) are widely used evaluation metrics for SR performance comparison, where $K = 5, 10, 20$ are the most common settings.

- **HR@K**. The HR@K measures whether the target item is included in the top-K recommendations in the recommended list.

$$\text{HR@K} = \frac{n_{hit}}{N} \tag{5.2}$$

[15] http://2016.recsyschallenge.com/

where N is the number of test sessions in the dataset and n_{hit} counts the number that target items that appear in the top K position of the ranking list.

- **MRR@K**. The MRR@K is a ranking evaluation matrix. When the target item $\hat{i}$ is not in the top K position, the $Rank(\hat{i})$ is set to 0. It calculates as follows,

$$\text{MRR@K} = \frac{1}{N} \sum_{\hat{i} \in \mathscr{S}_{test}} \frac{1}{\text{Rank}(\hat{i})} \tag{5.3}$$

where $\mathscr{S}_{test}$ are the set of test sessions. Rank($\hat{i}$) is the position of item i in the recommendation list. The MRR is a normalized ranking of hits. The higher the score, the better the recommendation quality.

- **NDCG@K**. The NDCG estimates the ranking order of the recommendation list. The same as MRR, if the target item $\hat{i}$ is not in the top K position, the $Rank(\hat{i})$ is set to 0.

$$\text{NDCG@K} = \frac{1}{N} \sum_{\hat{i} \in \mathscr{S}_{test}} \frac{1}{log_2(\text{Rank}(\hat{i}) + 1)} \tag{5.4}$$

Evaluation Metrics for Diversity. Diversification is first concerned in the information retrieval (IR) community, where researchers endeavor to disambiguate the input query to cover the user's real intent via diversification optimization [42]. In a recommendation system, diversity is related to how different the recommended items are from each other [17], which aims to alleviate the filter bubble problem. Consequently, the recommendation diversity can be identified as the average pairwise dissimilarity between items in the list. In SR, intra-list diversity, coverage, and their variants are the most common metrics for diversity measurement [150, 223, 279].

- **ILD**. Intra-list distance (ILD) measures the average distance between every pair of items in the recommendation list (RL), which can be formalized as,

$$\text{ILD} = \frac{\sum_{(i,j) \in RL} d_{ij}}{|RL| \times (|RL| - 1)} \tag{5.5}$$

where d_{ij} is the Euclidean distance of the category embeddings of item i and item j. |RL| is the number of recommendation items in RL.

- **Coverage@K**. Coverage@K measures how many different categories appear in the top-K recommendation items.

$$\text{Coverage@K} = \frac{\cap_{i \in RL_K} C_i}{K} \tag{5.6}$$

where C_i is the category of item i.

There are other metrics than the ILD coverage, such as long-tail coverage and relevance-sensitive expected intra-list diversity (RR-ILD). The former measures the diversity performance in long-tail items, the latter simultaneously considers the ranks and relevance of top K recommendations in ILD calculation.

5.6 Discussions

Although GNNs and sequential neural networks have greatly advanced SR research, they face several challenges. Consequently, we outline the following prospective research directions, which are critical for the SRs further development.

5.6.1 More External Information

Many efforts endeavor to fuse more external information to explore users' multi-interests or items' complicated transaction patterns for SR. However, current approaches necessitate the careful selection of external information to enhance performance. Besides, the fusion method is also significant for SR. To sum up, we consider that there are two open issues concerning external information that deserve further discussion.

- *What kind of external information is necessary for SR?* We divide the external information into three categories: item-based, interaction-based, and position-based. Therefore, some of them are costly to model (*e.g.,* item descriptions and user comments), some are not appropriate for SR (*e.g.,* user personal information as the sessions are expected to be anonymous in SR), some may not be necessary (*e.g.,* position or order information [195], since the length of sessions is limited and there is no clear order dependency between two items). In addition, the useful information may also vary across different recommendation scenarios. For instance, for news, book, or music recommendations, users will be interested in the categories of the items. In contrast, for product recommendations, the item's brand and price probability are more effective for SR. Hence, it is important to identify the valuable external information relevant to practical scenarios.

- *How to balance the trade-off between effectiveness and efficiency in fusion methods?* The fusion methods can be various in SR, *e.g.*, embedding addition or concatenation, self-attention, and gate mechanism. Note that as a straightforward manner for information fusion, the addition operation will not change the model structure

and increase numerous external parameters and computations. Nonetheless, this approach fails to thoroughly exploit external information, thereby limiting the model's representation ability. Although concatenation could enhance the representation ability, it will also dramatically increase the computational complexity and the number of parameters. In addition, concatenation cannot model the implicit correlation between any two external information. Recently, self-attention has been established as an effective method for external information fusion [266], though it also demands extra parameters and computational costs. Therefore, elegant fusion methods are expected to balance the trade-off between the representation ability and computational cost for an effective and efficient recommendation.

- *Couple vs. Decouple, which is better for SR?* In general, there are three phases for external information fusion. (1) Fusion first. Most of the studies consider fusing the external information in the embedding stage. Thus, the external information and item embedding will couple together for SR. (2) Fusion in process. GNN-based methods recognize the external information as nodes. Hence, a heterogeneous graph is constructed, and the external information can be fused based on information propagation and aggregation. (3) Fusion last. The external information and item embedding are decoupled and modeled first before being fused. For instance, we could construct two graphs for both item and external information, capitalizing on two GNNs for item and external information representation learning, which will be fused together in the end. Fusing first and then modeling the external information with a unified framework will limit the capacity of models to extract implicit features encapsulated in the heterogeneous side information, but it has a greater advantage in efficiency. In contrast, by decoupling external information representation with item representation and fusing later, we could fully exploit the features from different priorities and avoid mixed correlation effects. However, the computation cost might be unacceptable in a practical scenario.

5.6.2 Session Selection and Graph Construction

As the length of sessions is quite limited, *i.e.,* less than six for most public datasets, many studies introduce neighbor sessions for graph construction. Therefore, sophisticated transaction patterns can be captured for SR. Some issues need to be discussed for neighbor session selection and graph construction.

- *Scalability Graphs in SR.* Graph structure is pivotal for GNN-based SR. Large-scale and complicated graphs are constructed to harness the fertile information from neighbor sessions, including billions of nodes and edges in practical scenarios. Besides, each node contains a variety of features. Hence, straightforwardly applying such a huge graph for information propagation and prediction is nontrivial. To address this problem, sampling is a widely adopted solution *e.g.,* Graphsage [73]) and PinSage [282]. However, these methods are accompanied by high randomness, which may incur unstable model training. Consequently, how to design a scalable graph structure is vital for SR.

- *Dynamic Graphs in SR.* In the real world, the items and their relations are changing over time. Graph structures should be adjusted and updated dynamically to maintain up-to-date recommendations. Most of the existing studies are based on a static graph structure, few studies pay attention to dynamic graphs. Thus, it is a largely under-explored realm and deserves further study.

- *Self-learning Graph Structure.* Obtaining a proper graph structure requires considerable effort, and this process is also heuristic and problem-specific. Moreover, although recent research [32, 136] has revealed the necessity of modeling the implicit connections between items for SR, most of the existing graph-based methods can only capture the item relations with a few hops, which cannot explore the implication relations between items thoughtfully. Applying the self-learning strategy to graph construction is a promising approach to alleviating the above problems, as showcased in many other tasks [54, 120, 259]. In Chapter 6, we propose a self-learning graph structure to model the implicit relationships among nodes for session-based recommendation.

5.6.3 Diverse and Uncertain Representation of User Interests

Despite the superior performance of existing methods in SR, most of the efforts concentrate on a single and fixed user interest representation for recommendation and fail to disentangle the multiple interests of users. Considering the diversity of user interests, a fixed representation is insufficient and leads to sub-optimal results. In addition, users' sequential behaviors are uncertain. Therefore, compared with using a single fixed representation, it is worthwhile to capture interest diversity, inject uncertainties, and provide more flexibility for SR.

- *Diverse Representation of User Interests.* Rather than learning a single and fixed user interests representation, some studies aim to extend such a one-fold vector to multiple vectors with capsule networks or attention mechanism [191] for multi-interest, short-term and long-term interest, and interest diversity representation. For instance, Tian *et al.* [219] apply GCN and capsule network to capture a user's multi-level and multi-interest representation for recommendation. Guo *et al.* [68] propose a Hierarchical Leaping Network (HLN), which extracts various subsequences from the current session. The user's multi-interest representation can be captured by learning the representation of each subsequence. Li *et al.* [118] split the item embedding into several chunks and apply GGNN for each chunk to learn user interests representation with multiple factors for SR. [221] uses clustering technology for similar products to improve the diversity of recommendations. Aside from multi-interest representation, Li *et al.* [138] believe the user interest can be split into interest trend and interest diversity. The former is determined by her education level, income, and occupation, while the latter is easily impacted by advertisements or marketing. Therefore, the authors tailored two modules for user interest trends and diversity modeling. Although some studies focus on multiple interests representation, it is still in a preliminary stage, and many issues (*e.g.*, how to determine the number of interests in an adaptive fashion; how to fuse those interests representations for SR) need to be further explored.

- *Uncertain Representation of User Interests.* Compared with vector representation, distribution representation could inject uncertainties and provide more flexibility, it has been attracting interest from the research community [10, 77, 213, 226]. For instance, the item embedding can be initialized as a multi-Gaussian distribution governed by a mean vector and a covariance vector. The mean vector could reveal users' basic preferences, while the covariance vector injects a potential uncertainty. Fan *et al.* elaborate to apply distribution representation and *Wasserstein Distance* for recommendation [59, 60]. However, there are very few studies focused on this issue for SR. It will be interesting and valuable to explore uncertain representation in SR.

5.6.4 Explainability and Privacy Production for SR

Apart from the accuracy, outputs' explainability, security, and privacy production are also significant and expected for a good recommendation system. In general, the explanation

of a recommendation aims to answer ‚Äúwhy‚Äù, that is, why the items are recommended. An explainable recommendation can improve the transparency and persuasiveness of systems, boosting the satisfaction and stickiness of users [295]. With the prosperity of deep learning, sequential and graph-based neural networks behave as black boxes, making this research more challenging. Investigating the recent efforts, the efforts in this research area can be divided into two categories: (1) the explanation generation based on language models [20, 65, 124, 125, 146, 270] knowledge graphs [63, 264] or image visualizations [33], and (2) the explainability of deep learning model structures [33, 297]. However, very few studies [160, 255] focus on the explainability of SR. As for privacy production in recommendation, the user's historical interaction records encompass privacy information such as gender, age, and even political orientation, which can be inferred by the recommender system [314]. It is inappropriate to request such private, specific historical interactions for recommendation. Recently, attempts have resorted to unlearning strategies that eradicate the sensitive data in model training to tackle this problem [12, 267]. Overall, it is an interesting and promising topic for future research in SR.

5.6.5 Streaming or Online SR

As discussed in 5.6.2, in a real-life situation, sessions are dynamically produced as a stream, while most of the relevant work elaborates on training the recommendation system with the historical sessions, which preserve the users' static interests. Applying a static model for new sessions might be irrational as users' preferences change over time. Therefore, it is a challenge to effectively learn users‚Äô dynamic and real-time preferences for better SR. Existing research [70, 177] maintains a reservoir to update the model for an online recommendation. Specifically, the reservoir can be the incoming sessions containing new items or new users. Then, for each session, a sample probability is generated via Wasserstein distance (also known as Earth Mover's Distance [190]) or ranking-based distance [239]. After that, the sampling distribution is applied to update the reservoir. Thus, the model can be updated promptly based on the updated reservoir. However, the above methods heavily rely on the sampling strategy and distance computation, which is also expensive, making it difficult to balance the trade-off between effectiveness and efficiency. To overcome this issue, some studies [130, 262, 263] propose lightweight model architectures based on model compression techniques, such as low-rank decomposition, hash coding, and quantization. However, these solutions require a pre-defined fixed compressed ratio to retrain, leading to sub-optimal results when the

ratio is inappropriate. Effectively capturing users' dynamic preference changes to make real-time recommendations is a promising direction.

5.6.6 Causal Debias and Denoise in SR

Due to the exposure mechanism, popularity effects, and the feedback loop in the recommendation system, bias and noise problems become serious and heavily deteriorate the recommendation effectiveness [21]. To explain the relations between a cause and effect, causality-based methods, *e.g.*, causal inference, and causal graph, are a major solution to provide explanations and debias for recommendation system [145, 237, 250, 269, 273, 283, 296, 304]. Although there are limited studies specializing in this issue on SR, the causal model will bring SR research into a new frontier.

5.6.7 Reinforcement Learning for SR

Unlike supervised and unsupervised learning, reinforcement learning [108] focuses on goal-directed learning that maximizes the total reward achieved by an agent when interacting with its environment. Hence, it is a potential and prominent solution to model interactions between the user and agent, capture rapid changes in users' preferences, and realize dynamic recommendations. Recent years have witnessed significant progress of reinforcement learning in recommendation systems [35, 144]. Specifically, we could maintain and update a corresponding recommendation policy by modeling the user's interaction behaviors as a real-time decision-making process. For instance, in [203], the authors treated music playlist generation as a language modeling process, Thus, an attention-based language model with the policy gradient is applied for recommendation. Wang *et al.* [233] propose a method named knowledge-guided reinforcement learning (KERL), which integrates knowledge graphs into reinforcement learning. Specifically, KERL adopts TransE [11] with the MLP layer to predict future knowledge of user preferences and recommendations. Overall, the studies with reinforcement learning for SR could be summarized as (1) simulating users' interaction in sessions for dynamic and timely recommendations, (2) capturing the interest shift of users, (3) filtering the noise in sessions, and (4) selecting the valuable items for SR. However, model-free deep reinforcement learning requires lots of samples, as the received state is not guaranteed to be useful. In SR, the sessions are very short, accompanied by an extremely large action space (*i.e.*, number of items); thus, it will require more high-quality samples to cover the

exploration space, which hinders the further development of reinforcement learning in SR.

5.6.8 SR with Language Model

Encouraged by the remarkable success of large language models (LLMs) in NLP, utilizing language models for recommendation has recently become cutting-edge. Prompt or in-context learning and parameter-efficient fine-tuning (PEFT) are prominent solutions in this venue. For instance, Wang and Lim [230] design different prompting strategies to investigate the performance of GPT-3 [109] for next-item prediction. Along this line of research, Hou *et al.* [88] devise various prompting templates and formalize the sequential recommendation as a conditional ranking task. TALLRec [6], applies LoRA [90] to effectively fine-tune LLaMA [220] on recommendation datasets. M6-Rec [50] obtains M6 [142], a visual-linguistic pre-trained model, as the backbone and proposes an improved prompt tuning, named option tuning, for task-specific parameter fine-tuning. Apart from that, to alleviate the hallucination and improve the quality of outputs, retrieval-augmented generation (RAG) technology has become ubiquitous in LLM [64], where the external knowledge can be retrieved as auxiliary information to guide the LLM for generation. In recommendation with LLM, based on the user's historically interacted items, RAG is employed as a retriever for candidate selection or reranking [53, 289]. [16, 43] show that RAG is facilitated to address the cold-start problem and enhance the diversity of recommendations. However, few efforts explore the application of LLM and RAG in SR, leaving this research direction to be cultivated.

Unlike the auto-regressive generation strategy applied by LLMs, diffusion models establish a new paradigm for generative tasks and achieve remarkable success across a broad spectrum of applications [37, 46, 134, 272]. Overall, the diffusion model can be split into two stages: a *diffusion* stage that aims to corrupt the original input as a Gaussian distribution and a *reverse* stage that aims to iteratively recover the data from a Gaussian noise conditioned on the input. Diffurec [134] is the first work that applies diffusion for sequential recommendation. Besides, some studies explore the diffusion model in CTR prediction [238], multi-scenario recommendation [242]. However, no work has investigated the performance of the diffusion model in SR. Besides, the discrete embedding space and the time cost in the reverse stage also impede its application in online recommendation. Despite the drawbacks, diffusion models represent an uphill research area and are promising for SR.

5.6.9 Diffusion Model in SR

As a new paradigm of generation method, diffusion models achieve remarkable success in a wide range of tasks, such as computer vision, natural language processing, speech synthesis recommendation systems [46, 134, 272]. Overall, the diffusion model can be split into two stages: the diffusion stage aims to corrupt the original input as a Gaussian distribution, and the reverse stage where to recover the data from a Gaussian noise iteratively conditioned on the input. Theoretical underpinnings demonstrate that the diffusion model can acquire better generation in both quality and diversity against GAN and VAE methods [83, 206]. Diffurec [134] is the first work that applies diffusion for sequential recommendation. Specifically, the authors consider user and item representation as a Gaussian distribution for user multi-interest and item multi-aspect modeling, and the diffusion process is incorporated for representation generation. Attributed to the diversity of the reverse process, the diversity of the recommendation results can also be enhanced. In addition, some works explore the diffusion model in CTR prediction [238], multi-scenario recommendation [242]. However, the discrete embedding space and the time cost in the reverse stage impede the widespread application of the diffusion model in online recommendation. The diffusion model is an uphill research area, and I believe it will have a promising future in SR.

5.7 Conclusion

In this Chapter, we provide a comprehensive overview of GNNs and sequential neural networks for session-based recommendation. We first introduce the preliminaries and different graph structures in SR for easy reading. Then, we summarize the main focusing problems for SR in recent works and the typical solutions to tackle these challenges. Besides, a systematic classification, unique features, and a detailed comparison of GNNs and sequential neural networks, including the key motivations, main methods, and datasets, for SR are proposed. Moreover, the categorization scheme of relevant studies and their characteristics is also introduced. The statistical characteristics of public real-world datasets and a comparison of representative modules in terms of complexity and performance are introduced. Finally, we outline the challenges and future directions in this research field.

CHAPTER 6

DUAL GRAPH NEURAL NETWORKS FOR SESSION-BASED RECOMMENDATION

6.1 Introduction[1]

Session-based recommendation focuses on capturing users' short-term interests from sessions rather than exploring their rich historical interactions or modeling users' long-term interests [234]. It has shown significant advantages in dynamic and real-time recommendations. Existing session-based recommendation methods are mainly sequence-based or graph-based. Sequence-based methods view items in a session as a sequence and predict the next item with which users may interact. For example, Markov Chain-based methods [188, 198] map the current session into a Markov Chain and then infer the user's next action solely based on the last item in the session. Deep learning models like RNN, GRU, and LSTM [80, 81, 179, 217] are increasingly applied to session-based recommendation, given their outstanding feature representation ability. Several studies, e.g., NARM [126], SHAN [281], STAMP [149], and Transformers4rec [52], further apply attention mechanisms to distinguish the importance of items and capture user intentions in session-based recommendation. Graph-based methods rely on graph structures to represent relationships among items and aggregate the auxiliary information to improve performance. The graphs can be *intra-session* or *inter-session*. The former only considers the item relations within a single session [30, 69, 168] while the latter considers more

[1]This Chapter is based on our published work: Exploiting explicit and implicit item relationships for session-based recommendation.

than one session in the same recommendation problem [246, 305].

Sequential models generally become ineffective when dealing with short sessions, which widely exist [29, 176]. For example, the average and median lengths of sessions in a popular session dataset, *Diginetica*[2] are only 4.80 and 4.00. The lengths are 3.97 and 3.00 for another dataset, *Yoochoose*[3]. Besides, recent studies reveal that the ordering relationship between items in sessions may not be as effective for recommendation [29, 176]. Owing to the property of item correlation and multi-hop contextual information modeling within sessions, graph neural networks emerge and achieve remarkable success in session-based recommendation. Specifically, graph-based methods rely on problem-specific designs of graph structures to achieve good performance. Although predefined graphs contain prior domain knowledge, they may be incomplete or even unavailable due to the existing graph-based methods' focus on modeling explicit dependencies while neglecting the implicit relationships, which are proven equally important [32, 257].

To close this gap, we aim to leverage the explicit dependencies and dynamic correlations among items as reflected by sessions simultaneously for an effective session-based recommendation. To this end, we propose a novel **D**ual **G**raph **N**eural **N**etwork (DGNN), which models explicit dependencies and implicit correlations between items separately for node representation and next-item prediction. Decoupling explicit dependencies and implicit correlations could provide extra flexibility for model training [266] and improve interpretability through visualization. Here, we use examples (Figure 6.1, detailed in Chapter 6.2) to illustrate the distinctions of our approach from previous approaches in the graph structure.

In a nutshell, we make the following contributions in this work:

- To the best of our knowledge, we are the first to decouple explicit and implicit relationships among items in a holistic approach for effective session-based recommendation.
- We propose an adaptive graph neural network (A-GNN) to capture the implicit correlations between items in a self-learning strategy. This allows the graph structures to dynamically change during model training to accommodate the evolution of users' preferences.
- We present a novel graph neural network with a single gate (SG-GNN) to harness the explicit ordering (or sequential dependencies) of items in an inter-session

[2] http://cikm2016.cs.iupui.edu/cikm-cup
[3] http://2015.recsyschallenge.com/challege.html

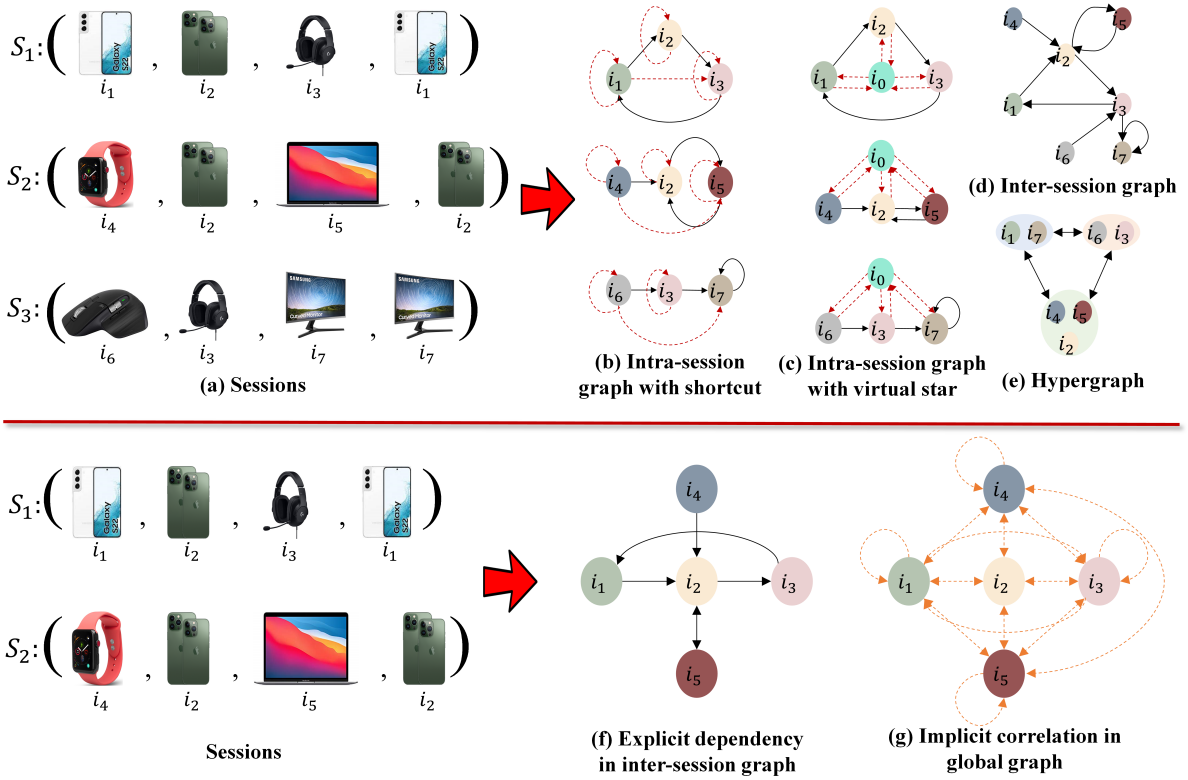


Figure 6.1: The upper half shows different graph structures for modeling item relationships in an example of three sessions. (b) applies shortcuts and self-loops (denoted by red dotted lines) to each session to capture long-range dependencies [30]; (c) creates a virtual item (i_0) to connect all the items in each session [168]; (d) illustrates all item relationships across all sessions with a single graph [268]; (e) groups items (e.g., according to their brands) and builds a hypergraph based on items' co-occurrence in the same sessions [261]. The lower half showcases our proposal of decoupling explicit and implicit item relationships for an example of two sessions.

graph.

- Our extensive experiments on four real-world datasets demonstrate that our model outperforms several baselines and state-of-the-art methods.

6.2 Problem Formulation

Let $\mathscr{I} = \{i_1, i_2, i_3, ...i_N\}$ be the set of items, where N is the number of items. Each session $s = [i_1, i_2, i_3, ..., i_o]$ consists of a sequence of interactions $i_k \in \mathscr{I} (1 \leq k \leq o)$ related to one user. Suppose we embed each item $i \in \mathscr{I}$ into the same space and denote by $\mathbf{x}_i \in \mathbb{R}^d$ the

representation of item i. Therefore, the representation of the item set is denoted by $\mathbf{X} \in \mathbb{R}^{N \times d}$.

Given a session s, session-based recommendation aims to predict the next click item $i_{s,m+1}$. Our model generates probabilities $\hat{\mathbf{y}}$ for all possible items based on the input session s. Each element's value of the vector $\hat{\mathbf{y}}$ is the recommendation score of the corresponding item. The items with the top-k recommendation scores will be recommended as the model's output.

Graph for explicit dependencies. As shown in Figure 6.1(f), given a session set $\mathscr{S}$, we denote the relationships between all adjacent items in $\mathscr{S}$ via an inter-session graph (dynamic global graph) $\mathscr{G}^s = (\mathscr{V}, \mathscr{E})$, where $\mathscr{V}$ and $\mathscr{E}$ indicate the set of nodes and edges, respectively. Each node represents an item in the session. For an edge $e \in \mathscr{E}$, it could be represented by an ordered tuple (v_i, v_j) (v_i, v_j are adjacent items in sessions) which indicates the edge points from node v_i to node v_j. Hence, we define an explicit dependency from v_i to v_j. The connectivity among the whole graph is represented by an adjacency matrix $\mathbf{A}^s \in \mathbb{R}^{N \times N}$ with $\mathbf{A}^s_{ij} \neq 0$ iff $(v_i, v_j) \in \mathscr{E}$ and $\mathbf{A}^s_{ij} = 0$ iff $(v_i, v_j) \notin \mathscr{E}$, where N is the total number of nodes. In addition, the adjacent matrix A^s is normalized following $\tilde{\mathbf{A}}^s = \mathbf{A_{ij}}^s / \sum_j \mathbf{A_{ij}}^s$. Hence, the inter-session graph could model the explicit dependency between items.

Graph for implicit correlations. As shown in Figure 6.1(g), given a session set $\mathscr{S}$, we could also construct a global graph $\mathscr{G}^g = (\mathscr{V}, \mathscr{E})$. We add an edge between any pair of nodes in $\mathscr{G}^g$ to indicate an implicit correlation. The element A^g_{ij} in an adjacency matrix $\mathbf{A}^g$ represents the correlation between item i and item j, which can be learned and adjusted dynamically by A-GNN. To decrease the time and space complexity, we construct a global graph for all the sessions in a batch, i.e., the sessions from the same batch will share the same global graph.

6.3 Methodology

The overall architecture of our proposed approach (Figure 7.2) consists of four major components: an adaptive graph neural network for implicit information aggregation and node representation (Chapter 6.3.2), a graph neural network with a single gate for explicit information aggregation and node representation (Chapter 6.3.3), a session representation layer (Chapter 6.3.4), a prediction layer and the loss function (Chapter 6.3.5).

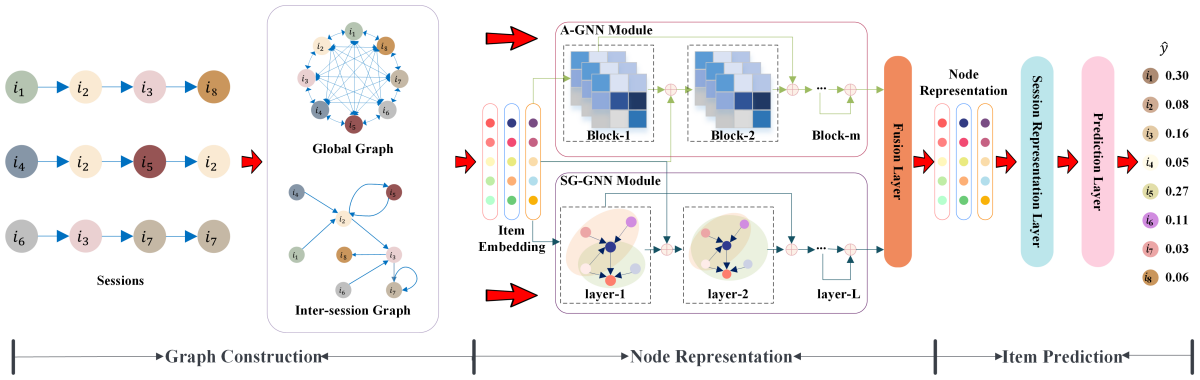


Figure 6.2: Architecture of DGNN.

6.3.1 Overview

Our framework works as follows. First, it constructs an inter-session graph. Specifically, we collect an item set based on neighbor sessions from one batch, and convert every item $v_i \in \mathcal{V}$ into a unified low-dimensional embedding space $\mathbf{X}$. Then, the item embedding is fed into a dual graph neural network (A-GNN and SG-GNN) to capture implicit and explicit item relationships. The fusion layer fuses the updated item representation $\tilde{\mathbf{X}}$ from those two modules. Finally, our soft-attention mechanism obtains session representations $\mathbf{s}$ and a *softmax* function generates the next item's prediction probability $\hat{\mathbf{y}}$. We formulate the above process as follows:

$$
\begin{aligned}
&\mathbf{X}^{(m)}_{\text{A-GNN}} = \text{A-GNN}(\mathbf{X} + \mathbf{X}^{(1)}_{\text{A-GNN}} + \ldots + \mathbf{X}^{(m-1)}_{\text{A-GNN}}, \mathbf{A}^{g}) \\
&\mathbf{X}^{(l)}_{\text{SG-GNN}} = \text{SG-GNN}(\mathbf{X}^{(l-1)}_{\text{SG-GNN}}, \tilde{\mathbf{A}}^{s}) \\
&\tilde{\mathbf{X}} = \text{F}(\mathbf{X}^{(m)}_{\text{A-GNN}}, \mathbf{X}^{(l)}_{\text{SG-GNN}}) \\
&\mathbf{s} = \text{SR}(\tilde{\mathbf{X}}) \\
&\hat{\mathbf{y}} = \text{P}(\mathbf{s}, \mathbf{X})
\end{aligned}
\tag{6.1}
$$

where $\mathbf{X}^{(m)}_{\text{A-GNN}}$ is the item representation of A-GNN with m blocks. $\mathbf{X}^{(l)}_{\text{SG-GNN}}$ is the item representation of SG-GNN after l convolution layers. $\text{F}(\cdot), \text{SR}(\cdot), \text{P}(\cdot)$ are the representation fusion layer, session representation layer, and prediction layer, respectively.

6.3.2 A-GNN Module

The adaptive graph neural network (A-GNN) module aims to capture implicit correlations between any of two items dynamically with a self-learning strategy for item

representation. To achieve this goal, A-GNN employs multi-head correlation, formulated below,

$$
\begin{aligned}
&\mathbf{Q}_i = \mathbf{X}\mathbf{W}_i^Q, \quad \mathbf{K}_i = \mathbf{X}\mathbf{W}_i^K, \quad \mathbf{V}_i = \mathbf{X}\mathbf{W}_i^V \\
&\mathbf{A}_i^g = \text{Dropout}(\tanh(\mathbf{Q}_i\mathbf{K}_i^T)) \\
&\mathbf{X}_{\text{A-GNN}_i} = \mathbf{A}_i^g\mathbf{V}_i \\
&\mathbf{X}_{\text{A-GNN}} = \text{Dropout}(\text{ReLU}([\mathbf{X}_{\text{A-GNN}_0}||...||\mathbf{X}_{\text{A-GNN}_k}])\mathbf{W}^M)
\end{aligned}
\tag{6.2}
$$

where $\mathbf{X} \in \mathbb{R}^{N\times d}$ is the item embedding, $\mathbf{X}_{\text{A-GNN}}$ is the item representation generated by A-GNN, and $\mathbf{A}^s$ is the adjacency matrix of the dynamic global graph. Each element of $\mathbf{A}^g$, say $\mathbf{A}^g_{ij}$, represents the correlation between item i and item j. $\mathbf{W}_i^Q, \mathbf{W}_i^k, \mathbf{W}_i^V, \mathbf{W}^M$ are all learnable parameter matrices. $||$ is a concatenation operation, and k is the head number (in this Chapter, we set $k = 4$). Compared with existing self-attention mechanisms [224], A-GNN replaces *softmax* with a *tanh* function to cope with non-positive correlations between items. It also differs from GAN [225] in obtaining the correlations between any pair of items rather than with the neighboring items.

We stack multiple A-GNN blocks to enhance the model's representation capacity. As such, each module takes all the previous blocks' outputs as input:

$$
\mathbf{X}^{(m)}_{\text{A-GNN}} = \text{A-GNN}(\mathbf{X} + \mathbf{X}^{(1)}_{\text{A-GNN}} + ... + \mathbf{X}^{(m-1)}_{\text{A-GNN}}, \mathbf{A}^g)
\tag{6.3}
$$

where A-GNN$(\cdot)$ denotes the A-GNN block, m is the number of A-GNN blocks, and $\mathbf{X}^{(i)}_{\text{A-GNN}}$ is the i-th block's output. A-GNN's final output is the representation of the last block M, i.e., $\mathbf{X}^{(M)}_{\text{A-GNN}}$.

6.3.3 SG-GNN Module and Fusion Layer

The graph neural network with a single gate (SG-GNN) module aims to leverage the explicit dependencies among items as reflected by the sequential information in sessions. To this end, SG-GNN aggregates the information of neighbors into the center node via a gate mechanism for representation update:

$$
\begin{aligned}
&\mathbf{J} = \mathbf{X}\mathbf{W}_J, \quad \mathbf{P} = \mathbf{X}\mathbf{W}_P, \quad \mathbf{Z} = \mathbf{X}\mathbf{W}_Z \\
&\mathbf{R} = \tilde{\mathbf{A}}^s\mathbf{J}\mathbf{W}_R, \quad \mathbf{U} = \tilde{\mathbf{A}}^s\mathbf{J}\mathbf{W}_U \\
&\mathbf{X}_{\text{SG-GNN}} = \mathbf{P} + \text{ReLU}(\mathbf{R} + \mathbf{Z}) \odot \mathbf{U}
\end{aligned}
\tag{6.4}
$$

where $\tilde{\mathbf{A}}^s$ is the normalized adjacency matrix (defined in Chapter 6.2). $\mathbf{W}_J$, $\mathbf{W}_R$, $\mathbf{W}_Z$, $\mathbf{W}_P$, $\mathbf{W}_U$ are learnable parameter matrices. The gate mechanism controls how much information from neighbors is considered for updating the node representation.

We apply multi-layer graph convolution as described below. In particular, we use item embedding $\mathbf{X}$ as the first layer's input. SG-GNN's final output is the last layer's item representation $\mathbf{X}^{(L)}_{\text{SG-GNN}}$.

$$\mathbf{X}^{(l)}_{\text{SG-GNN}} = \text{SG-GNN}(\mathbf{X}^{(l-1)}_{\text{SG-GNN}}, \tilde{\mathbf{A}}^s) \tag{6.5}$$

Given implicit and explicit representations of items from A-GNN and SG-GNN, we fuse them using a linear projection, thus obtaining the final item representation $\tilde{\mathbf{X}}$ as follows.

$$\tilde{\mathbf{X}} = [\mathbf{X}^{(M)}_{\text{A-GNN}} || \mathbf{X}^{(L)}_{\text{SC-GNN}}]\mathbf{W}_F \tag{6.6}$$

where $\mathbf{W}_F \in \mathbb{R}^{2d \times d}$ is a learnable parameter matrix.

6.3.4 Session Representation Layer

We use local and global representations of sessions to capture users' short-term and long-term preferences. Given a session $s = [i_1, i_2, ..., i_m]$, we assume users' current preference can be reflected by the last item i_m, following previous research [257]. We thereby use the representation of the last-clicked item i_m as the session's local representation, i.e., $\mathbf{s}_l = \tilde{\mathbf{x}}_m$. As for the global representation of sessions s, $\mathbf{s_g}$, we generate it based on the representations of all items in the session. Specifically, we employ a soft-attention mechanism to fuse the information from all items while taking into account their varied importance:

$$\begin{aligned} \alpha_i &= \mathbf{q}^T \sigma(\mathbf{W}_1 \tilde{\mathbf{x}}_m + \mathbf{W}_2 \tilde{\mathbf{x}}_i + \mathbf{c}) \\ \mathbf{s}_g &= \sum_{i=1}^{m} \alpha_i \tilde{\mathbf{x}}_i \end{aligned} \tag{6.7}$$

where $\mathbf{q}^T, \tilde{\mathbf{x}}_i \in \mathbb{R}^d$ and $\mathbf{W}_1, \mathbf{W}_2 \in \mathbb{R}^{d \times d}$ are all learnable parameters. α controls the weights of item representations.

Finally, we concatenate the local and global representations via a linear transformation to obtain the final session representation $\mathbf{s}$:

$$\mathbf{s} = [\mathbf{s}_l || \mathbf{s}_g]\mathbf{W}_3 \tag{6.8}$$

where matrix $\mathbf{W}_3 \in \mathbb{R}^{2d \times d}$ compresses two combined representation vectors into the latent space $\mathbb{R}^d$.

6.3.5 Prediction Layer and Loss Function

We calculate item scores $\hat{\mathbf{z}} \in \mathbb{R}^N$ as the inner production of item embedding $\mathbf{X}$ and session representation $\mathbf{s}$:

$$\hat{\mathbf{z}} = \mathbf{s}^T \mathbf{X} \tag{6.9}$$

Then, we apply a *softmax* function to the scores for next-item prediction. This will generate probabilities indicating how likely each item would be the next to be clicked by the user:

$$\hat{\mathbf{y}} = \text{softmax}(\hat{\mathbf{z}}) \tag{6.10}$$

For each session, we define the loss function as the cross-entropy of the prediction and the ground truth:

$$\mathscr{L}(\hat{\mathbf{y}}) = -\sum_{i=1}^{n} \mathbf{y}_i log(\hat{\mathbf{y}}_i) + (1 - \mathbf{y}_i) log(1 - \hat{\mathbf{y}}_i) \tag{6.11}$$

where $\mathbf{y}$ is the one-hot encoding of the ground-truth item.

6.4 Experiments

We report our experimental setting, including datasets, baselines, evaluation metrics, and an analysis of experimental results. We aim to answer the following questions:

- **RQ1.** How does the DGNN perform compared with state-of-the-art (SOTA) session-based recommendation methods?
- **RQ2.** How do different sub-modules in the DGNN affect recommendation performance?
- **RQ3.** How do hyperparameter settings influence model performance?
- **RQ4.** How is the model interpretation capability of DGNN?

6.4.1 Datasets

We conducted experiments on four real-world datasets commonly used for session-based recommendation.

Table 6.1: Statistics of datasets

	Diginetica	Yoochoose1/64	Yoochoose1/4	Gowalla	Last.FM
#clicks	981,620	557,248	8,326,407	1,122,788	3,835,706
#train sessions	716,835	369,859	5,917,745	675,561	2,837,644
#test sessions	60,194	55,898	55,898	155,332	672,519
#items	42,596	16,766	29,618	29,510	38,615
#length ≤ 5	537,546	289,490	4,234,915	627,100	1,136,909
#length >5	239,483	136,267	1,738,734	203,793	2,373,254
Average length	4.80	6.16	5.71	4.32	9.16

- *Diginetica* is a personalized e-commerce research challenge dataset from CIKM CUP 2016. The dataset contains transition history, which is suitable for session-based recommendation. Following [19, 30, 149, 185, 257], we used the sessions in the last week for testing.

- *Yoochoose* is a dataset that contains a stream of user clicks on an e-commerce website within six months from the RecSys Challenge 2015. We conducted the typical method in [19, 30, 149, 185, 240, 257] to split the dataset. Since the training set of Yoochoose is extremely large, we used the most recent portions, 1/64 and 1/4 subsamples of all training sessions as the training set, denoted as "Yoochoose1/64" and "Yoochoose1/4", respectively.

- *Gowalla* is a popular dataset widely used for point-of-interest recommendation. Following [19, 30, 69], we kept the top 30,000 most popular locations and grouped users' check-in records into disjoint sessions by splitting intervals between adjacent records that are longer than one day. We used the last 20% of sessions as the test set.

- *Last.FM* is a music artist recommendation dataset. Following [19, 30, 69, 185], we kept the top 40,000 most popular artists and treated users' transactions in 8 hours as a session. Like Gowalla, we used the most recent 20% of sessions as the test set.

Following [19, 30, 69, 185, 257], we filtered out sessions of length 1 and items appearing less than 5 times. Furthermore, for each given session $s = [i_1, i_2, ..., i_m]$, we generated the input and corresponding labels, i.e., $([i_1], i_2), ([i_1, i_2], i_3), ..., ([i_1, i_2, ..., i_{m-1}], i_m)$, for all the datasets. Table 6.1 summarizes the statistics for the datasets.

6.4.2 Baselines and Evaluation Metrics

We chose 11 baselines from five categories of methods: conventional methods (e.g., popularity- and Nearest Neighbors (NN)-based methods), sequence-based methods including Markov Chain and sequential neural networks, graph neural networks with intra-session graphs and inter-session graphs, as listed below:

- **POP** is a simple benchmark that recommends the most popular (highest ranked) item for users.
- **Item-KNN** [193] recommends items through the similarity between every item of the current session and the other items.
- **FPMC** [4] [188] combines the first-order Markov Chain with matrix factorization to capture both sequential effects and user preferences.
- **GRU4Rec**[5] [81] employs a gated recurrent unit to model the sequential behavior of items in a session.
- **NARM**[6] [126] improves GRU4Rec by introducing RNN with attention to session-based recommendation.
- **SR-GNN** [7] [257] models explicit dependencies within a session via a graph neural network and then applies a soft-attention mechanism to generate session-level embeddings.
- **SGNN-HN** [168] applies a star graph neural network to model the complex transition relationships between items without direct connections in an ongoing session.
- **LESSR** [8] [30] introduces two kinds of session graphs with self-loops and shortcuts to capture implicit connections and solve the information loss and long-range dependency problem.
- **MSGIFSR** [9] [69] proposes a consecutive intent unit to extract user intent from different granularities based on different item groups in the current session. It achieves the latest SOTA in the above four datasets.

[4] https://github.com/khesui/FPMC
[5] https://github.com/hidasib/GRU4Rec
[6] https://github.com/lijingsdu/sessionRec_NARM
[7] https://github.com/CRIPAC-DIG/SR-GNN
[8] https://github.com/twchen/lessr
[9] https://github.com/SpaceLearner/SessionRec-pytorch

- **GC-SAN** [268] gets local context information by using GGNN and then utilizes a self-attention mechanism to capture explicit dependency.

- **GCE-GNN** [246] considers ϵ-neighbor ($\epsilon = 2$) connections to construct an inter-session graph for session-based recommendation.

We evaluated all models with two widely used metrics: HR@20 (Hit Rate) and MRR@20 (Mean Reciprocal Rank). HR@20 represents the proportion of correctly recommended items among the top 20 items. MRR@20 is the average reciprocal rank of the correctly-recommended items. The reciprocal rank is set to 0 when the rank exceeds 20.

6.4.3 Experimental Setup

For a fair comparison, we followed [69, 168, 257] and selected the Adam optimizer with the initial learning rate of 0.001, which will decay by 0.5 after every five epochs. We set the L_2 regularization to 10^{-5} and used an early stopping strategy (no improvements in the evaluation metrics for five consecutive epochs) to relieve the overfitting problem. We initialized all parameters using a Gaussian distribution with a mean of 0 and a standard deviation of 0.1. We fixed both the embedding dimension and batch size at 100. For the GC-GNN module, the number of layers varied within {1,2,3,4}. For the A-GNN module, the block number was within the scope of {4,5,6}. We tested the dropout ratio within {0.1, 0.5, 0.9}.

6.4.4 Overall Comparison (RQ1)

Our comparison results (Table 7.1) show our method (DGNN) significantly outperformed all the baselines, which is largely attributed to the two modules' capability to capture more accurate and complete user preferences—while SG-GNN can effectively integrate the explicit information from neighbors through the improved graph convolution operation, the self-learning dynamic graph can learn implicit correlations between items, which are equally important for improving the recommendation performance. In particular, DGNN outperformed state-of-the-art (SOTA) performance by a large margin, i.e., a 115.29% improvement, on *Last.FM*, which contains longer sessions when compared with other datasets. This reveals the ability of graph neural networks to handle prediction tasks on long-range sessions when equipped with explicit and implicit item relationship modeling.

Table 6.2: Experimental results (%) on the four datasets. The best results are highlighted in boldface, and the second-best results are underlined. * denotes a significant improvement of DGNN over the best baseline results (t-test P<.05).

Model	Diginetica		Yoochoose 1/64		Yoochoose 1/4		Gowalla		Last.FM	
	HR@20	MRR@20	HR@20	MRR@20	HR@20	MRR@20	HR@20	MRR@20	HR@20	MRR@20
POP	0.89	0.28	6.71	1.65	1.37	0.31	1.46	0.38	5.26	1.26
Item-KNN	37.75	11.57	51.60	21.81	52.31	21.70	38.60	16.66	14.90	4.04
FPMC	26.53	6.66	45.62	15.01	51.86	17.50	29.91	11.45	12.86	3.78
GRU4Rec	29.45	8.22	60.64	22.89	59.53	22.60	41.98	18.37	17.90	5.39
NARM	49.70	16.00	68.32	28.63	69.73	29.23	50.07	23.92	21.83	7.59
SR-GNN	50.73	17.78	70.57	30.94	71.36	31.89	50.32	24.25	22.33	8.23
SGNN-HN	55.67	19.45	72.13	32.60	73.52	32.63	55.28	27.58	25.07	9.40
LESSR	51.71	18.15	70.59	31.46	72.67	33.12	51.34	25.49	23.37	9.01
MSGIFSR	57.11	20.05	73.13	33.50	74.01	33.74	56.64	29.02	27.63	10.86
GC-SAN	51.70	17.61	70.66	30.04	71.83	30.93	50.68	24.67	22.64	8.42
GCE-GNN	54.02	19.04	70.91	30.63	71.40	31.49	53.96	24.53	24.39	8.63
DGNN	**67.65***	**27.89***	**75.85***	**34.09***	**76.90***	**36.02***	**58.51***	**30.40***	**47.17***	**23.38***
Improv.	18.46%	49.10%	3.72%	1.76%	3.90%	6.76%	3.30%	4.76%	70.72%	115.29%

Deep learning methods performed significantly better than traditional methods (e.g., POP, Item-KNN, and FPMC), demonstrating their superior complex feature extraction and representation ability. NARM outperformed GRU4Rec because NARM can not only capture the latent sequential information in sessions (as GRU4Rec does) but also learn item correlations via the attention mechanism. GNN-based models generally outperformed sequence-based methods, showing the importance of session graphs in representing transition relationships between different items. MSGIFSR designs various granular intent units to model the implicit and multi-granular relationships among items, thus achieving the latest SOTA for session-based recommendation. This suggests the necessity and significance of designing sophisticated modules to capture implicit correlations between items for session-based recommendation.

6.4.5 Ablation Study (RQ2)

To verify the effectiveness of A-GNN and SG-GNN in DGNN, we removed or replaced one of these modules from DGNN to analyze the performance change.

- **MLP-SR**: replaces A-GNN and SG-GNN with one MLP layer with *ReLU* activation function. The session representation layer, prediction layer, and loss function remain the same as in DGNN.
- **w/o SG-GNN**: removes SG-GNN from DGNN.
- **w/o $\sum$**: removes the accumulated operation in A-GNN and only feeds the output of the latest block into the next block for implicit correlation modeling.

Table 6.3: Results (%) of ablation experiments.

Datasets		MLP-SR	w/o A-GNN	w/o SG-GNN	w/o $\sum$	w Self-Att	w GGNN	DGNN
Diginetica	HR@20	58.60	53.26	49.67	50.54	50.83	64.22	**67.65**
	MRR@20	20.77	17.71	16.50	17.44	16.54	25.14	**27.89**
Yoo 1/64	HR@20	70.07	71.28	68.10	73.23	70.68	69.25	**75.85**
	MRR@20	30.53	30.87	28.68	32.21	30.84	28.90	**34.09**
Yoo 1/4	HR@20	70.23	74.97	69.88	69.45	75.24	76.64	**76.90**
	MRR@20	31.02	33.41	30.32	30.78	33.38	35.99	**36.02**
Gowalla	HR@20	51.73	52.62	49.47	55.31	50.70	**59.27**	58.51
	MRR@20	25.12	25.63	23.75	26.59	24.37	28.85	**30.40**
LastFM	HR@20	21.98	23.89	21.06	23.39	23.21	38.12	**47.17**
	MRR@20	8.13	9.09	7.74	9.02	8.91	17.47	**23.38**

Table 6.4: Time and Space Complexity. We set the size of learnable parameter matrices to the dimension of the item embedding d, and the size of graphs to $N \times N$ for SG-GNN and GGNN.

Module	Number of Parameters	FLOPs
GGNN	$d \times (11d + 8)$	$2d \times (2N + 11d)$
SG-GNN	$d \times 5d$	$2d \times (2N + 5d)$

- **w Self-Att**: replaces A-GNN with a multi-head self-attention module [224] and only uses the representation from the last block for implicit correlation modeling.
- **w GGNN**: replaces SG-GNN with the GGNN module in SR-GNN [133, 257].

Our results (Table 7.2) show that MLP-SR beats all sequential models on all datasets except *LastFM*, which contains much longer sessions than other datasets. The reason lies in that a naive MLP layer could be sufficient for capturing the global information from shorter sessions, while sequential models might be more efficient in handling longer sessions. All modules were shown to be effective, given that removing any of them would drastically degrade the performance.

DGNN outperformed many baselines (e.g., SR-GNN, LESSR, GC-SAN) even without A-GNN, which is impressive, considering SG-GNN only contains half the number of parameters and floating point operations (FLOPs) in SG-GNN (refer to Table 6.4). The performance of DGNN significantly decreased when a self-attention module replaced A-GNN. But changing the information aggregation modules (SG-GNN to GGNN) did not notably impact the results. This indicates that A-GNN is robust to the graph neural networks for explicit dependency modeling as an auxiliary module for session-based

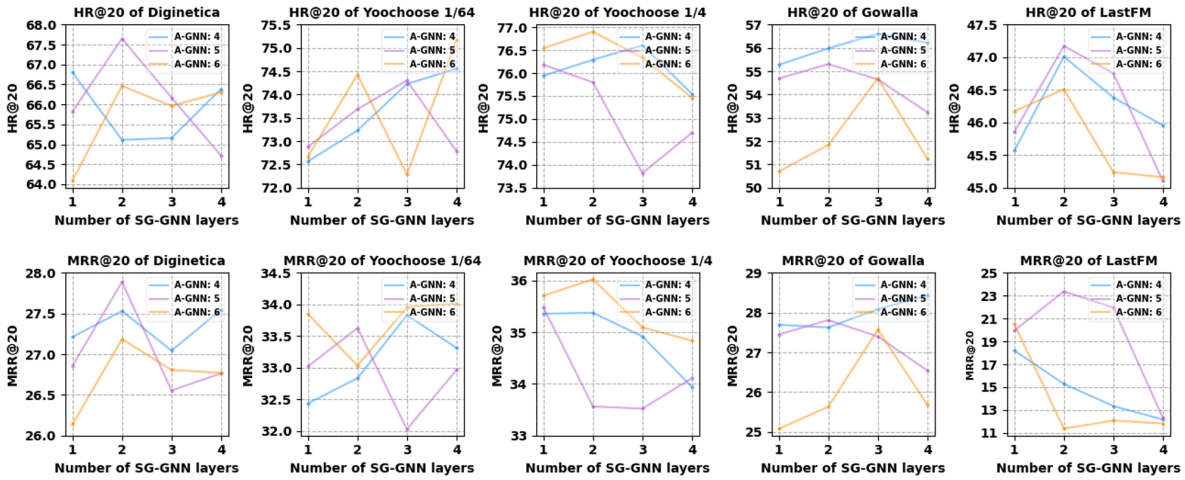


Figure 6.3: Parameter sensitivity of the number of A-GNN blocks and IP-GNN layers.

recommendation. After removing the accumulated operation in A-GNN (w/o $\sum$), we observed a significant drop in the performance of DGNN. But still, A-GNN beats self-attention even without the accumulation operation. Therefore, we conclude both the output of previous blocks in A-GNN and the *tanh* activation function are critical to our approach.

6.4.6 Impact of Hyper-parameter Setting (RQ3)

We studied two parameters, the number of A-GNN blocks and the number of SG-GNN layers, in this experiment. Our results (Figure 6.3) show that a moderate number (neither too large nor too small) of A-GNN blocks generally resulted in better performance. An exception is with the *Yoochoose* dataset, which is extremely large and contains complex transaction patterns. Since the sampled sub-datasets (namely *Yoochoose 1/64* and *Yoochoose 1/4*) cannot fully cover the (original) entire dataset's features, more A-GNN blocks and SG-GNN layers enhance the feature extraction capability to model such complex transaction patterns. Since SG-GNN can obtain the global information from the whole dynamic graph with one convolution operation, a larger number of multiple graph convolution layers will result in smoother item representations, which could negatively affect DGNN's performance.

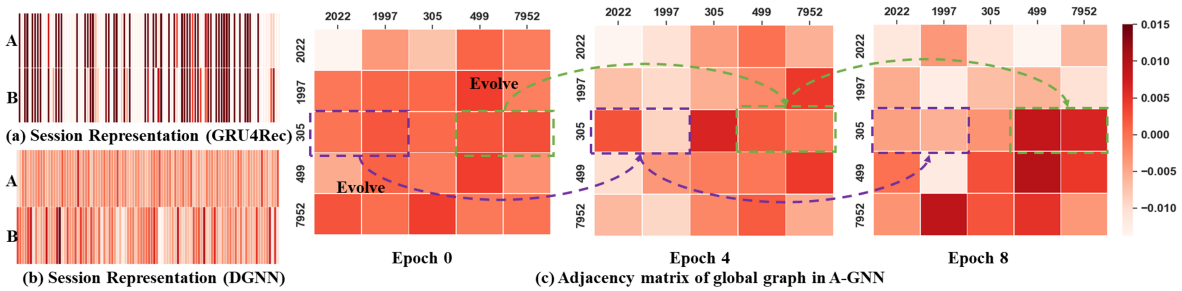


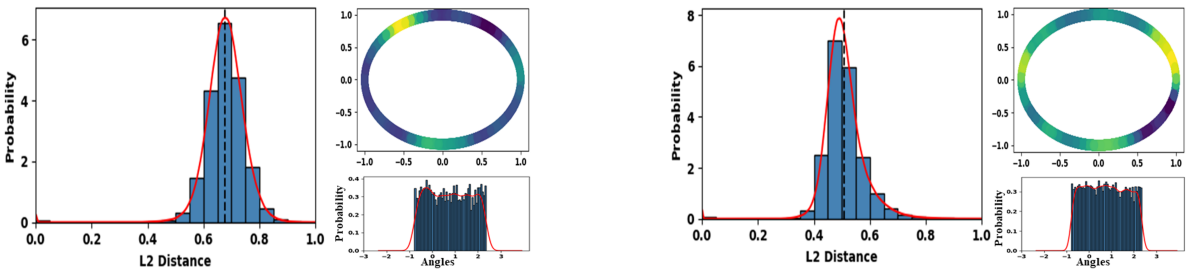
Figure 6.4: (a) and (b) are the representations of session A:{7951, 7952, 4999, 7952, 305} and session B:{4999, 7951, 7952, 305, 7952} generated by GRU4Rec and DGNN. (c) visualizes the adjacency matrices in A-GNN at epochs 0, 4, and 8, respectively.

6.4.7 Visual Analysis of A-GNN (RQ4)

Session representation. We retrieved two random sessions from the *diginetica* dataset to explore the impact of sequence information on session representation. The two sessions contained the same items in a different order. We selected a typical sequential neural network, GRU4Rec, as well as our method, DGNN, to generate the session representations. Our results (Figure 6.4a-b) show that GRU4Rec generated similar representations for session A and session B. These representations, however, significantly differ from the representations output by DGNN. It suggests that sequential neural networks, e.g., GRU4Rec, may not capture the sequential information in sessions as effectively as our GNN-based approach.

Implicit Correlation. We visualized the global graph adjacency matrices in A-GNN at Epoch 0, Epoch 4, and Epoch 8, to better understand the evolution of implicit correlation between items during the training process. Our results (Figure 6.4 c) revealed that the correlation between any two items was similar at the beginning (Epoch 0). As the training progressed, the correlation between item 305 (ground truth) and 499, 7952 (previously interacted items) increased while the correlation between 305 and 2022, 1997 (negative samples randomly selected from the item set) dropped steadily. The above results demonstrate that the adjacency matrix in A-GNN can successfully distinguish positive implicit correlations from weak or negative ones between items. This validates our assumption of the existence of implicit correlations between items, with the correlations between items within the same sessions tending to be positive and those between irrelevant items being negative.

Item Representation. We visualized the item representations generated by a self-attention module and A-GNN to offer further insights into the superior effectiveness



(a) Self-attention Module (b) A-GNN Module

Figure 6.5: Item representations of (a) self-attention and (b) A-GNN on $\mathscr{S}^1$ (two-dimensional space). **Alignment analysis**: the histograms show the distributions of l_2 distance between the representations of item pairs, where the black dotted lines indicate the mean distances. **Uniformity analysis**: the other plots in subfigures show the distributions of item representations with Gaussian kernel density estimation (KDE) in $\mathbb{R}^2$ (top-right) and with von Mises-Fisher (vMF) KDE on angles (bottom-right), i.e., $\arctan2(y,x)$ for each point $(x,y) \in \mathscr{S}^1$. The darker the color, the denser the distribution in the top-right plots. Item representations generated by A-GNN are more *aligned* (lower l_2 distances) and *uniform* (evenly distributed).

of A-GNN over the self-attention module in implicit correlation modeling. We consider two key properties in contrastive learning [236] for our visualization task: (1) *alignment* (closeness) of item representations from item pairs; (2) *uniformity* of the induced distribution of the (normalized) item representations on the hypersphere. We randomly selected 5,000 item representations generated by the self-attention module and A-GNN for the *Diginetica* dataset, respectively. Then, we calculated the l_2 distance of any two items to plot the frequency distribution histogram. We further visualized the normalized item representation distribution with Gaussian kernel density estimation (KDE) in $\mathbb{R}^2$. Figure 6.5 shows the above results. Comparing the l_2 distance distributions of item pairs' representations obtained by the self-attention module (the histogram in Figure 6.5a) and A-GNN module (the histogram in Figure 6.5b), we observed that A-GNN resulted in a smaller mean distance (black dotted line) than self-attention, indicating the item representations generated by A-GNN were more closely clustered. A comparison of other diagrams in Figure 6.5 suggests that A-GNN could obtain a more uniform item representation distribution on $\mathscr{S}^1$. The above analysis implies that a uniform and aligned distribution of item representations could benefit session-based recommendation.

6.5 Conclusion

In this work, we propose to decouple the modeling of explicit dependencies and implicit correlations among items for session-based recommendation. We present a dual graph neural network (DGNN), where a GNN with a single gate (SG-GNN) captures the explicit dependencies as reflected by the ordering of items in sessions, and an adaptive GNN (A-GNN) learns implicit correlations between any two items adaptively with a self-learning strategy. Our extensive experiments demonstrate the superiority of DGNN to SOTA on four public datasets. Besides, A-GNN is shown to generate a more uniform and aligned distribution of item representations. As we split the batch in a random manner, the adaptive graph will be different in each training process. Thus, the results are somewhat unstable. In future research, we will explore how to construct an effective and efficient adaptive graph for robust performance.

CHAPTER 7

TAIL ITEMS IN SEQUENTIAL RECOMMENDATION

7.1 Introduction[1]

Sequential recommendation aims to recommend the next item the user prefers based on the historical interaction sequences. Early studies focus on the Markov chain method, which models a user behavior sequence as a Markov Decision Process (MDP) for next-item prediction [188, 198]. Given the promising results of deep neural networks in a variety of tasks, many studies seek to apply GRU, LSTM, Transformer, and their variants to sequential recommendation [81, 103, 214]. Among those techniques, graph neural networks, diffusion methods, and contrastive learning have attracted increasing attention in sequential recommendation research [128, 134, 257, 265]. Recently, natural language processing (NLP) and computer vision (CV) applications have witnessed the great success of large pre-training models. This inspires emerging studies on applying LLMs or LVMs to sequential recommendation [66, 87]. As examples, UnisRec [87] and Recformer [123] follow a similar procedure: they first pre-train a language model via item text (e.g., title, categories, brands) on source domains and then fine-tune it on target domain data for making recommendations. P5 [65] is another paradigm that defines the sequential recommendation as a next-token generation task; it fine-tunes LLMs with prompt engineering for next-item prediction. Based on P5, VIP5 [66] further incorporates image representation as the embedding for sequential recommendation. For image-

[1]This Chapter is based on our published work: Reembedding and Reweighting are Needed for Tail Item Sequential Recommendation.

based or hybrid sequential recommendation (which leverages both text and images), the majority of studies [66, 76, 252] adopt LLMs or LVMs to obtain items' text and image representations. Afterwards, a dedicated cross-attention module is employed for representation fusion. In view of the immense computational cost and memory footprint of LLMs-based or LVMs-based recommenders, MMMLP [140] only applies MLP layers as the backbone for sequential recommendation. Other studies use tailored adapters for parameter-efficient fine-tuning and recommendations [6, 143].

Extensive studies have shown that the abundant external knowledge encapsulated in the LLMs or LVMs could enhance item representations for better performance. However, those large models suffer more significant performance degradation in tail item recommendation against ID-based methods, and this issue is largely overlooked in existing research, hindering their applicability to long-tailed data. As an example, Figure 7.1 (left) shows the highly skewed and long-tailed item distribution of *Amazon* dataset, where 80% item set has fewer than 17 user interactions and only 5% items have more than 50 interactions. Figure 7.1 (right) further compares the performance of image-based [2] and ID-based [3] recommendations, showing the image-based model performs poorly on tail items despite its superior overall performance to the ID-based model.

We establish that existing LVMs/LLMs-based models face significant challenges preserving high performance on tail items in sequential recommendation, i.e., only very few head items receive vast attention while the majority of items (a.k.a., tail items) are unpopular and attract very limited interactions. Firstly, the CE loss focuses exclusively on increasing the likelihood of the target item (ground-truth) while treating all non-ground-truth items as equally incorrect [85, 156], i.e., *all-in ground-truth*. However, user preferences may vary across items and inherently require differentiated optimization efforts; moreover, tail items might be sufficiently optimized when compared with popular items; further, some non-ground-truth items might still be preferred by users, provided they are exposed to the users. The insufficient optimization of tail items is reflected in Figure 7.1 (left), where the average embeddings of popular items (with more interactions) are more distinct and sharper; in contrast, the average embeddings of tail items (with very few interactions) tend to be uniform. Secondly, although the item representations derived from LLMs/LVMs encompass enriched knowledge, such prior knowledge

[2]We use the item image representation generated by CLIP (`https://huggingface.co/openai/clip-vit-large-patch14`.) as the initial item embedding and train a Transformer-based model with the cross-entropy loss for recommendation.

[3]We use item ID as embedding and train a Transformer-based model with the cross-entropy loss for recommendation.

encapsulated in these representations could dominate optimization directions. Since recommendation models rely more on the knowledge of LLMs and LVMs than on the collaborative filtering signals learned from historical records, such external knowledge transfer may adversely impact the models, causing performance degradation on tail items, i.e., *knowledge transfer tax*.

The tail item problem widely and chronically exists in online services, incurring popularity basis and impairing recommendation performance. Existing efforts to tackle this problem mainly focus on introducing auxiliary information to enhance item representations, especially tail item presentations, for sequential recommendation. Tyipcal auxiliary information include assistance relationships from head-item [98, 106, 152], similar items [271] or similar sequences [92], and semantic information from LLMs [148], LVMs [9] or knowledge graph (KG) [299]. Despite the above efforts, the improvement is rather limited for image-based recommendation, leaving the issue underexplored. Moreover, none of them provide a theoretical discussion to investigate the underlying causes or propose a solution to address the issue.

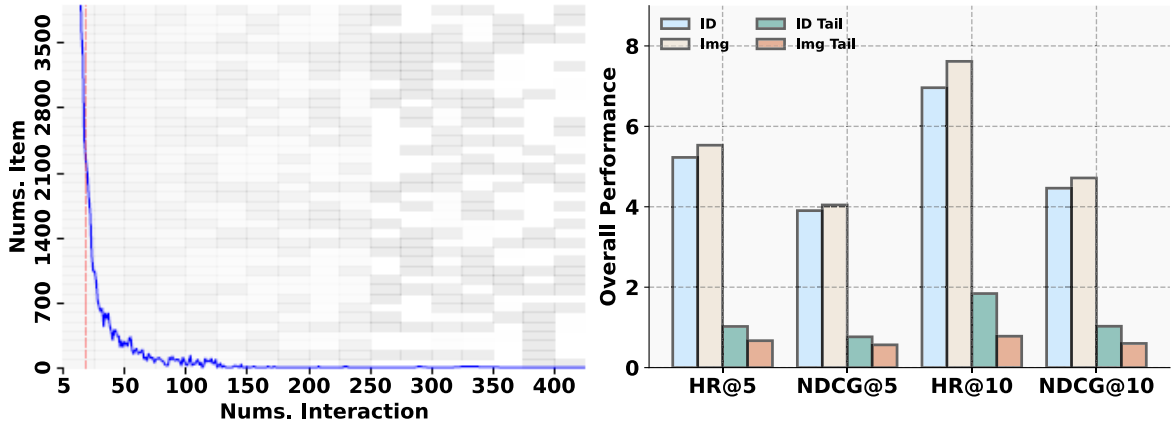


Figure 7.1: The left part illustrates the long-tailed distribution of items in the *Amazon* dataset, where the blocks in the background represent the average embedding of items with different interaction counts. It shows 80% items have fewer than 17 interactions (to the left of the red dashed line), and tail item embedding possesses a more uniform-like distribution due to insufficient training. The right part shows that the image-based model achieves superior overall performance but performs worse than the ID-based model on tail items.

To address the above issues, we propose R^2Rec, a simple yet efficient approach based on reweighting and reembedding for the sequential recommendation. Specifically, we first reinitialize the tail item embedding using a standard Gaussian distribution to avoid

the negative effect of external knowledge transfer. Then, we incorporate a reweighting function in the vanilla cross-entropy (CE) loss. The function adaptively adjusts score distribution during training, pushing the model towards paying more attention to tail items. As such, it considers both recommendation accuracy and diversity on tail items, enforcing nuanced optimization and alleviating the issue caused by insufficient training on tail items. This function formulation is potentially comparable to the preference alignment algorithms (e.g., RLHF [167], DPO [183]), which are designed to guide the model toward recommending tail items in LLMs.

In a nutshell, we make the following contributions in this work:

- We first explore the *all-in ground-truth* and *knowledge transfer tax* issues with LLM/LVM-based recommenders. We also provide a mathematical analysis to investigate the performance degradation of LLM/LVM-based recommenders on tail items from those two perspectives.

- We propose a simple yet effective approach, R^{2}Rec,, which reinitializes tail item embedding and reweights the CE loss adaptively during model training to address the above-identified issues.

- We conducted extensive experiments on three real-world datasets to demonstrate the advantages of our approach in promoting tail item preferences. The results show our proposed method outperforms all the 14 baselines and improves the recommendation performance on tail items by a large margin.

7.2 Methodology

7.2.1 Problem Statement

Sequential Recommendation. Let the user set and item set be $u \in \mathscr{U}$ and $i \in \mathscr{I}$, respectively. Given a chronologically organised sequence of historically interacted items of user u, i.e., $s_u = [i_1, i_2, ..., i_\ell]$, sequential recommendation aims to predict the probability that user u will be interested in item i at the next step $\ell+1$, i.e., $P(i_{\ell+1}|s) = Q_\theta(i_{\ell+1}|s)$, where $Q_\theta(\cdot)$ is a sequential recommendation model parameterized by θ.

The predominant method to estimate θ is to train model Q_θ on a vast corpus of historical interaction sequences using maximum likelihood estimation. Given the target item distribution P, the objective of model training is to minimize the cross-entropy

between P and Q_θ, as formalized below:

$$\mathscr{L}_{\mathrm{CE}}(P, Q_\theta) = -\mathbb{E}_{i_{\ell+1}\sim P}[\log Q_\theta(i_{\ell+1}|s)] \tag{7.1}$$

7.2.2 Deficiency of CE Loss and Text-based or Image-based Embedding

All-in Ground-truth. Following the standard CE loss specified in Eq. (7.1), we derive the gradient cross-entropy with regard to model parameters θ as follows:

$$\nabla_\theta \mathscr{L}_{\mathrm{CE}}(P, Q_\theta) = -\mathbb{E}_{i_{\ell+1}\sim P}[\frac{\nabla_\theta \log Q_\theta(i_{\ell+1}|s)}{\log Q_\theta(i_{\ell+1}|s)}] \tag{7.2}$$

According to Eq. (7.2), when minimizing CE loss via gradient descent, Q_θ is encouraged to assign a high probability to the target item (ground-truth), i.e., high likelihood items under P shall also have high likelihood under Q_θ [85, 186]. In contrast, the remaining non-ground-truth items will not receive any explicit activation for optimization during the training. It is both undesirable and unreasonable in practice: firstly, the non-interacted items might still be a potential target, i.e., it may just be that they haven't been exposed to the user in the past; secondly, not all non-ground-truth items should be considered equal. Ideally, the training should reward potential candidates by increasing their probabilities to varying degrees rather than penalize them by reducing their probabilities to zero. These deficiencies in the CE loss hinder the model from achieving accurate and diverse recommendations [135].

Knowledge Transfer Tax. When the training process is sufficient, the external knowledge encapsulated in image/text-based embedding could enhance item representations, improving recommendation performance. For tail items, however, owing to the limited interaction records, the prior knowledge in embeddings may dominate the training process and optimization directions of the model, inducing performance degradation. Specifically, given the historical sequence s, let $P(i|s)$ be the predicted probability distribution of item i, which can be estimated by the model $Q_\theta(i|s)$, i.e., $P(i|s) = Q_\theta(i|s)$. Suppose $P_c(i|s)$ is the probability distribution of item i predicted based on the knowledge from LLMs or

LVMs. We have

$$
\begin{aligned}
P(i|s) &= \frac{P_c(i|s) \cdot \frac{P(i|s)}{P_c(i|s)}}{\sum_{k \in \mathcal{I}} P_c(k|s) \cdot \frac{P(k|s)}{P_c(k|s)}} \\
&= \frac{P_c(i|s) \cdot \frac{P(s|i)}{P_c(s|i)} \cdot \frac{P(i)}{P_c(i)} \cdot \frac{P_c(s)}{P(s)}}{\sum_{k \in \mathcal{I}} P_c(k|s) \cdot \frac{P(s|k)}{P_c(s|k)} \cdot \frac{P(k)}{P_c(k)} \cdot \frac{P_c(s)}{P(s)}} \\
&= \frac{P_c(i|s) \cdot \frac{P(s|i)}{P_c(s|i)} \cdot \frac{P(i)}{P_c(i)}}{\sum_{k \in \mathcal{I}} P_c(k|s) \cdot \frac{P(s|k)}{P_c(s|k)} \cdot \frac{P(k)}{P_c(k)}} = \frac{\zeta_c(s) P(i) P(s|i)}{\sum_{k \in \mathcal{I}} \zeta_c(s) P(k) P(s|k)}
\end{aligned}
\tag{7.3}
$$

Deep neural networks typically apply a *softmax* function to the negative log-likelihood to obtain the optimization objective during model training. As such, we could derive the loss function from Eq. (7.3) and Eq. (7.1) as follows:

$$
\begin{aligned}
&\mathscr{L}_{CE}(P, Q_\theta) = \mathscr{L}_{CE}(P, P(i|s)) = -\mathbb{E}_{i \sim P}[\log P(i|s)] \\
&= -\mathbb{E}_{i \sim P}[\log \frac{\exp(\zeta_c(s) + P(i) + P(s|i))}{\sum_{k \in \mathcal{I}} \exp(\zeta_c(s) + P(k) + P(s|k))}]
\end{aligned}
\tag{7.4}
$$

where $\zeta_c(s) = \frac{1}{P_c(s)}$. Generally, this term does not significantly impact the training process as the external knowledge from the pre-trained embedding contains limited information relevant to the sequence s. However, in tail-item prediction, the values of $P(i)$ and $P(s|i)$ are significantly smaller due to the lower occurrence of tail items in the dataset, causing $\zeta_c(s)$ to dominate the loss function. In this case, the model will rely more on the prior knowledge from LLMs or LVMs instead of the collaborative patterns learned from sequences for recommendation, which, in turn, impairs the recommendation performance on tail items.

7.2.3 R^{2}Rec, Framework

The framework of our proposed R^{2}Rec, is depicted in Figure 7.2, which consists of three main components: (1) the image-based transformer backbone for a sequential recommendation, (2) the reweighting function incorporated with the CE loss and (3) the embedding (incl., reembedding) operation applied on tail items.

Image-based Recommendation. In R^{2}Rec„ we adopt the vanilla Transformer [224] as our backbone. For the prediction layer, we apply a linear projection operation that leverages the updated representation of the sequence‚Äôs final item, which encapsulates the information of the entire sequence, to predict the next item for recommendation. We obtain the corresponding image of the input item and apply the CLIP [5] image encoder

[5] `https://huggingface.co/openai/clip-vit-large-patch14`

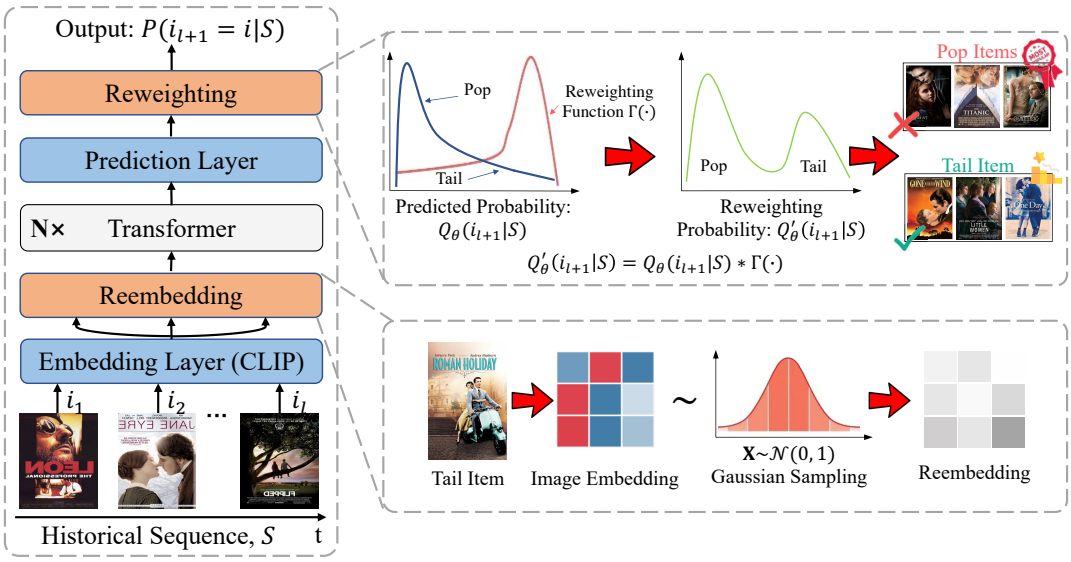


Figure 7.2: Framework of R^{2}Rec,. It adopts Transformer as the backbone (left) and includes two key modules (right), i.e., Reembedding and Reweighting.

(CLIP-Img) [182] as the embedding layer for item representation initialization. It is formalized as follows:

$$\begin{aligned} &\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_\ell = \text{CLIP-Img}(i_1, i_2, ..., i_\ell) \\ &\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_\ell = \text{ImgRec}(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_\ell) \\ &\mathbf{y}_{\ell+1} = \mathbf{W}\mathbf{h}_\ell^T \end{aligned} \tag{7.5}$$

where $[i_1, ..., i_\ell]$ and $[\mathbf{x}_1, ..., \mathbf{x}_\ell], \mathbf{x}_k \in \mathbb{R}^{1\times d}$ are the item image and the correspondent image-based embedding generated by the CLIP. $[\mathbf{h}_1, .., \mathbf{h}_\ell], \mathbf{h}_k \in \mathbb{R}^{1\times d}$ are the updated item representation generated by the transformer. $\mathbf{y}_{\ell+1} \in \mathbb{R}^{N\times 1}$ is the predicted target item scores at step $\ell + 1$, N is the total number of items. $\mathbf{W} \in \mathbb{R}^{N\times d}$ is a learnable parameter matrix. The predicted scores will generally undergo a *softmax* function to obtain the probability and then be fed into the CE loss for model optimization.

Reembedding Operation. Following Eq. (7.4), the external knowledge encapsulated in the tail item embeddings will dominate the model optimization directions and hurt the model performance. Consequently, a straightforward recipe is to reinitialize the text-based or image-based tail item embeddings as a standard Gaussian distribution (i.e., $\mathbf{x} \sim \mathcal{N}(0,1)$) and train them from scratch. This process can be formalized below,

$$\mathbf{x}_i = \begin{cases} \text{CLIP-Img}(i) & \text{if } i \notin \mathcal{I}_{TL} \\ \sim \mathcal{N}(0,1) & \text{otherwise} \end{cases} \tag{7.6}$$

Reweighting Function. As discussed in Chapter 7.2.2, the standard CE loss treats all the items as equal. This is inappropriate as the tail items are the minority in the training samples and can not receive sufficient training. Consequently, we propose an efficient reweighting function that adaptively adjusts the predicted probability, optimizing the training process to pay more attention to the tail items. Formally, the CE loss incorporated with the reweighting term can be defined as follows:

$$\begin{aligned}\mathscr{L}_{\mathrm{RCE}}(P,Q_\theta) &= -\mathbb{E}_{i_{\ell+1}\sim P}[\log f(Q_\theta(i_{\ell+1}|s))] \\ &= -\mathbb{E}_{i_{\ell+1}\sim P}[\log Q_\theta(i_{\ell+1}|s)\Gamma(i_{\ell+1},Q_\theta(i_{\ell+1}|s))] \\ &= -\sum_{i\in\mathscr{I}} P(i_{\ell+1}|s)\log Q_\theta(i_{\ell+1}|s)\Gamma(i_{\ell+1},Q_\theta(i_{\ell+1}|s)) \\ s.t \quad \sum_{i\in\mathscr{I}} Q_\theta(i_{\ell+1}|s) = 1 \quad \forall s\in\mathscr{S}; \quad &\sum_{i\in\mathscr{I}} \Gamma(i_{\ell+1},Q_\theta(i_{\ell+1}|s)) = 1\end{aligned} \tag{7.7}$$

where $\Gamma(\cdot)\in\mathbb{R}^{N\times N}$ is the reweighting function. Intuitively, the reweighting function is required to (1) be the function of Q_θ; (2) assign larger weight to the tail items and in turn decrease the weight of the head items and (3) the weight distribution should be adjusted dynamically during the model training, i.e., with different weights applied at each epoch. Therefore, we formalize the reweighting function $\Gamma_j(i,Q_\theta(\hat{i}|s))$ at j-th training epoch as:

$$\Gamma_j(i,Q_\theta(\hat{i}|s)) = \frac{\exp(\eta_j(i,Q_\theta(\hat{i}|s))/\tau)}{\sum_{i\in\mathscr{I}}\exp(\eta_j(i,Q_\theta(\hat{i}))/\tau)} \tag{7.8}$$

$$\eta_j(i,Q_\theta(\hat{i}|s)) = \begin{cases} \eta_{j-1}(i,Q_\theta(\hat{i}|s)) + \alpha_p & \text{if } i\in\{\hat{i}_1^{j-1},...,\hat{i}_k^{j-1}\} \text{ and } i\in\mathscr{I}_{TL} \\ \eta_{j-1}(i,Q_\theta(\hat{i}|s)) - \alpha_r & \text{if } i\notin\{\hat{i}_1^{j-1},...,\hat{i}_k^{j-1}\} \text{ and } i\in\mathscr{I}_{TL} \\ \alpha_b & \text{others} \end{cases} \tag{7.9}$$

where i is the ground-truth and $\{\hat{i}_1^{j-1},...,\hat{i}_k^{j-1}\}$ are the top-K item list at the $(j-1)$-th training epoch predicted by the model Q_θ. We set $K=5$. $\mathscr{I}_{TL}$ are the tail item set. $\alpha_p,\alpha_r,\alpha_b$ are the plenty, reward, and base factors. We set $\alpha_p=\alpha_b=1$ and $\alpha_r=0$. τ is a temperature factor to control the shape of the distribution, we set $\tau=0.5$. We initialize the $\eta_0=1$ at the beginning of the training epoch.

Training Process. The full training process is summarized in Algorithm 1. Compared to the standard training process, we introduce only two additional operations: reembedding at the first item embedding layer and reweighting at the final loss function calculation stage. Therefore, our method can be easily adapted to a wide range of recommendation models with minimal modifications.

Algorithm 1: R^{2}Rec, Traning Process

Input:
Image-based historical sequence: $[i_1, \cdots, i_\ell]$;
Ground-truth probability: $P_{\ell+1}$;
Tail item set: $\mathscr{I}_{TL}$;
CLIP image encoder: CLIP-Img$(\cdot)$;
Transformer-based backbone ImgRec: $Q_\theta(\cdot)$;
Reweighting function: $\Gamma(\cdot)$;
Learning epochs: T;
Optimizer: AdamW$(\cdot)$;
Output:
Predicted target item: $\hat{i}_{\ell+1}$;
while $j < T$ **do**
$[\mathbf{x}_1, ..., \mathbf{x}_\ell]$ = CLIP-Img$(i_1, .., i_\ell)$; `// Embedding initialization`
$\mathbf{x}_j \sim \mathcal{N}(0,1), i_j \in \mathscr{I}_{TL}$;`// Reembedding`
$\mathscr{L}_{RCE}(Q_\theta([\mathbf{x}_1, ..., \mathbf{x}_\ell]), P, \Gamma)$;`// RCE loss, Eq. (7.7) to 7.9`
$\theta \leftarrow$ AdamW$(\mathscr{L}_{RCE}, \theta)$; `// Parameter update`
$j = j + 1$;
end while

7.2.4 Discussion

We discuss the properties of the proposed reweighting function, providing insights into its advantages over the standard cross-entropy loss, particularly with respect to tail items during optimization.

More Precise Optimization. The CE loss is demonstrated to be the ground-truth first, as it treats all non-ground-truth equally, providing no supervised signal for their optimization. To amend this problem, we proposed RCE loss, incorporating a reweighting function to CE loss. Considering the gradient of general RCE formalizes with respect to model parameters θ, we have:

$$\begin{aligned} \nabla_\theta \mathscr{L}_{\text{RCE}}(P, Q_\theta) &= -\mathbb{E}_{i_{\ell+1} \sim P}[\log \nabla_\theta Q_\theta(i_{\ell+1}|s) \Gamma(i_{\ell+1}, Q_\theta(i_{\ell+1}|s))] \\ &= -\sum_{i \in \mathscr{I}} \log \nabla_\theta Q_\theta(i_{\ell+1}|s) \mathbb{E}_{i_{\ell+1} \sim P}[\Gamma(i_{\ell+1}, Q_\theta(i_{\ell+1}|s))] \end{aligned} \tag{7.10}$$

From Eq. (7.10), we can observe that the model gradients ∇_θ will be updated under the supervision of the reweighting function $\Gamma(\cdot)$, which is defined across all the items. Therefore, all the items will participate and contribute to the model optimization process. By designing different $\Gamma(\cdot)$, we can modulate the attention assignment to items. In

contrast to the CE loss, RCE endows a more nuanced optimization due to the availability of diverse supervisory weights for all items.

Connection with Direct Preference Optimization (DPO). DPO is one of the most popular offline preference optimization methods used for preference alignment in LLMs [183]. Instead of learning an explicit reward model [167], DPO algorithm optimizes the policy in a straightforward manner by reparameterizing the reward function $r(\cdot)$ using a closed-form expression in a supervised manner:

$$r(x,y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{(ref)}(y|x)} + \beta \log Z(x) \tag{7.11}$$

where $\pi_\theta(\cdot|x)$ and $\pi_{\text{ref}}(\cdot|x)$ are the policy model and reference model, respectively. β is the coefficient of the partition function or the normalizing constant $Z(x)$.

By incorporating the reward function (Eq. (7.11)) into the Bradley-Terry (BT) ranking objective formula [14],

$$p(y_w \succ y_l|x) = \frac{1}{1+\exp(r(y_w) - r(y_\ell))} = \sigma(r(x,y_w) - r(x,y_\ell)) \tag{7.12}$$

We can cancel out the partition function $Z(x)$, resulting in the objective of DPO with reverse KL divergence below:

$$-\mathbb{E}_{(x,y_w,y_\ell)\sim\mathscr{D}} \left[\log\sigma \left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_\ell|x)}{\pi_{\text{ref}}(y_\ell|x)} \right) \right] \tag{7.13}$$

where $\sigma(\cdot)$ is the sigmoid function. y_w and y_ℓ are preference pairs consisting of the approved (win) response and refused (lose) response with regard to the input x.

Based on the DPO function, we define $\beta = 1$, thus, the Eq. (7.13) can be expressed below:

$$\begin{aligned} &-\mathbb{E}_{(x,y_w,y_\ell)\sim\mathscr{D}} \left[\log\sigma \left(\log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \log \frac{\pi_\theta(y_\ell|x)}{\pi_{\text{ref}}(y_\ell|x)} \right) \right] \\ &= -\mathbb{E}_{(x,y_w,y_\ell)\sim\mathscr{D}} \left[\log\sigma \left(\log \frac{1}{\pi_\theta(y_\ell|x)} * \pi_\theta(y_w|x) \right) \right] \end{aligned} \tag{7.14}$$

As the original DPO only models the pairwise preference comparison (i.e., $\pi_\theta(y_w|x)$ and $\pi_\theta(y_\ell|x)$), instead, we consider all the possible outputs and therefore replace $1/\pi_\theta(y_\ell|x)$ as the reweighting function $\Gamma(y, \pi_\theta(y|x))$, which can also control the preference alignment based on the plenty or reward factors. Moreover, we replace the σ as *softmax* function. Therefore, the Eq. (7.14) can be rewritten into

$$\begin{aligned} &-\mathbb{E}_{(x,y)\sim\mathscr{D}} \left[\log \frac{\pi_\theta(y_i|x)\gamma(y_i, \pi_\theta(y_i|x))}{\sum_{y_j\in\mathscr{Y}} \pi_\theta(y_j|x)\gamma(y_j, \pi_\theta(y_j|x))} \right] \\ &= -\mathbb{E}_{y\sim P}[\log Q_\theta(y|x)\Gamma(y, \pi_\theta(y|x))] \end{aligned} \tag{7.15}$$

From this perspective, our reweighting function can be recognized as a preference alignment strategy, steering the model to prioritize tail item outputs.

Pre-trained knowledge from different domains could facilitate the model performance via transfer learning. Given a sequence s, the target item of prediction is i, we define the probability that the model can make the correct prediction based on the knowledge from k-th source domain is $P(f_k(i|s)) = 1-\epsilon$, where ϵ is the probability of errors. Assuming each source domain is independent and the error probabilities are the same, then we have,

$$P(H(n) \leq m) = \sum_{i=0}^{m} C_n^i (1-\epsilon)^i \epsilon^{n-i} \tag{7.16}$$

where $H(n)$ is the number of source domains that can share auxiliary knowledge to assist the model in making the correct prediction. n is the total number of source domains. Based on the Hoeffding's inequality [84], we have,

$$P(H(n) \leq (p-\delta)n) \leq \exp(-2\delta^2 n) \tag{7.17}$$

Let $\delta = \frac{(1-2\epsilon)}{2}, m = \frac{n}{2}$, we substitute Equation 7.17 with Equation 7.16,

$$P(H(n) \leq n/2) \leq \exp(-\frac{1}{2} n(1-2\epsilon)^2) \tag{7.18}$$

Note that as the number of source domains (i.e., n) increases, the probability that more than half of the domains contribute useful information to aid the model in making accurate predictions increases. Therefore, we believe the transferred knowledge from cross-domains can enhance the model's performance in downstream tasks.

7.3 Experiments

We conduct extensive experiments over three real-world datasets against fourteen baselines for performance evaluation. Overall, we aim to answer the following questions:

- **RQ1.** How does the overall performance of R^{2}Rec, and its performance on tail items compare with state-of-the-art sequential recommendation methods?
- **RQ2.** How do tailored reweight and reembedding operations applied in R^{2}Rec, affect its performance?

- **RQ3.** How do different hyperparameter settings affect the performance of R^2Rec, on three datasets?
- **RQ4.** How do item embeddings evolve throughout the various phases of model training?

7.3.1 Datasets and Evaluation Metrics

Datasets. We select three subcategories, i.e., *Beauty*, *Toys*, and *Sports*, from Amazon datasets[6] [161] for performance evaluation, as they contain enriched item information (e.g., item title, description, and image, and so on) and user information (e.g., user ratings and reviews, and so on). All of these datasets are widely used in sequential recommendation. Following previous works [9, 66, 103], we recognize the user-item ratings as interactions, and the items and users with fewer than five interaction records are removed. For each user, we organize the filtered interactions chronologically based on the timestamp and adopt the leave-one-out strategy for the training dataset, validation dataset, and testing dataset split, i.e., given a sequence $s = [i_1, i_2, ..., i_{\ell+1}]$, we gather the most recent interaction $i_{\ell+1}$ for model testing, the penultimate interaction i_ℓ for model validation, and the remains $[i_1, i_2, ..., i_{\ell-1}]$ for model training. We set the maximum sequence length as 10 for all the datasets. Based on the Pareto principle [13] as the criteria, we set the ratio as 20% to curate the tail item.

Evaluation Metrics. We apply two popular recommendation metrics, HR@N (Hit Rate) and NDCG@N (Normalized Discounted Cumulative Gain), for performance evaluation. The HR@N measures how many hits are present within the top-N recommended list, which reveals the capability of models in recall. NDCG@N further evaluates the model ranking performance by considering the ranking position of these hits in the list. We set N = 5 and N = 10 to compare the experimental results of our R^2Rec, with the baseline models. Due to the negative sampling evaluation will incur a distinct gap against the practical scenario when the size of negative samples is small [111], we take all the items as candidates for performance comparison.

7.3.2 Baselines and Implementation Details

Baselines. We select three categories of methods, including (1) convention ID-based models; (2) image-based models, text-based models, and image-text-based models; and (3)

[6] https://nijianmo.github.io/amazon/index.html

tail-item-oriented methods, for a comprehensive performance evaluation. These methods are highly relevant to our research.

- **ID-based Models.** We select **GRU4Rec**[7] [81], **Caser**[8] [218], **SASRec**[9] [103], **S^3-Rec**[10] [310], the four representative ID-based methods for performance evaluation, covering three mainstream neural network architectures, i.e., GRU, CNN, and Transformer.

- **Text-based and Image-based Models.** We include two text-based models (i.e., **UniSRec**[11] [87] and **P5**[12] [65]), two image-based models (i.e., **MM-Rec**$^{-}$[13] [252] and **MMMLP**$^{-}$[14] [140]) and three multi-modality (image and text) models (i.e., **MM-Rec**, **MMSBR**[15] [293], and **MMMLP**). All of them are committed to obtaining the item image or text representation via vision and language models and then dedicating representation fusion modules (e.g., cross-attention) for recommendation.

- **Tail Item Oriented Models.** We compare our R^2Rec, against three tail item sequential recommendation solutions, i.e., **CITES** [98], **MELT**[16] [106], and **MAN** [141]. To tackle the challenge of tail item recommendation, all of them endeavor to introduce more auxiliary information from, e.g., head items, context items, and cross-domains, for representation enhancement.

Implementation Details. We apply the AdamW optimizer with the learning rate of $5e-4$ and adopt the warm-up strategy with a step ratio of 0.1. We set the multi-head numbers as 16 and the block numbers as 1. Furthermore, we set the hidden size of Transformer blocks as 768, the same as the embedding size. The dropout ratio is 0.8, and the batch size is 128. As for the hyperparameter τ in the reweighting function, we set $\tau = 0.5$ and further analyze the efficiency within the scope of 0.1 to 1. We compare

[7] https://github.com/hidasib/GRU4Rec
[8] https://github.com/graytowne/caser
[9] https://github.com/kang205/SASRec
[10] https://github.com/RUCAIBox/CIKM2020-S3Rec
[11] https://github.com/RUCAIBox/UniSRec
[12] https://github.com/jeykigung/P5
[13] "-" means the text modeling module is removed from raw models.
[14] https://github.com/Applied-Machine-Learning-Lab/MMMLP
[15] https://github.com/Zhang-xiaokun/MMSBR
[16] https://github.com/rlqja1107/MELT

Table 7.1: Overall performance on the three datasets. The best results are highlighted in boldface, and the second-best results are underlined. "−" means the text modeling module is removed from raw models and only uses image information for recommendation. -txt and -img denote item text, and item image information is considered, respectively, for embedding initialization and recommendation. † means cross-domain transfer learning is applied. ▲% means improvement (%) against the best results excluding the R^2Rec, variants. * denotes a significant improvement over the best baseline results (t-test P<.05).

Dataset	Amazon Beauty				Amazon Toys				Amazon Sports			
	HR@5	HR@10	NDCG@5	NDCG@10	HR@5	HR@10	NDCG@5	NDCG@10	HR@5	HR@10	NDCG@5	NDCG@10
GRU4Rec	0.0164	0.0283	0.0099	0.0137	0.0097	0.0176	0.0059	0.0084	0.0129	0.0204	0.0086	0.0110
Caser	0.0232	0.0394	0.0149	0.0201	0.0202	0.0329	0.0121	0.0162	0.0130	0.0222	0.0079	0.0108
SASRec	0.0327	0.0626	0.0240	0.0323	0.0454	0.0655	0.0301	0.0375	0.0172	0.0325	0.0089	0.0138
S^3-Rec	0.0387	0.0647	0.0244	0.0327	0.0443	0.0700	0.0294	0.0376	0.0251	0.0385	0.0161	0.0204
UniSRec	0.0476	0.0734	0.0263	0.0331	0.0455	0.0713	0.0254	0.0337	0.0264	0.0457	0.0143	0.0220
P5	0.0494	0.0690	0.0394	0.0412	0.0619	0.0716	0.0312	0.0425	0.0290	0.0381	0.0168	0.0215
MM-Rec$^-$	0.0377	0.0546	0.0224	0.0279	0.0596	0.0779	0.0336	0.0405	0.0279	0.0404	0.0162	0.0201
MMMLP$^-$	0.0313	0.0494	0.0211	0.0269	0.0215	0.0338	0.0147	0.0187	0.0157	0.0267	0.0100	0.0136
MMSBR	0.0331	0.0557	0.0201	0.0273	0.0299	0.0476	0.0183	0.0240	0.0182	0.0318	0.0116	0.0160
MMMLP	0.0526	0.0754	0.0382	0.0448	0.0588	0.0812	0.0436	0.0488	0.0320	0.0448	0.0219	0.0263
MM-Rec	0.0381	0.0552	0.0227	0.0282	0.0603	0.0783	0.0338	0.0414	0.0294	0.0419	0.0174	0.0215
MELT	0.0221	0.0428	0.0118	0.0184	0.0284	0.0491	0.0144	0.0211	0.0170	0.0289	0.0103	0.0141
CITIES	0.0487	0.0695	0.0355	0.0422	0.0570	0.0751	0.0426	0.0484	0.0278	0.0417	0.0190	0.0235
MAN	0.0535	0.0715	0.0398	0.0456	0.0606	0.0769	0.0449	0.0502	0.0311	0.0430	0.0223	0.0262
R^2Rec,-txt	0.0538	0.0739	0.0398	0.0462	0.0622	0.0807	0.0464	0.0524	0.0287	0.0384	0.0204	0.0235
R^2Rec,-img	0.0545	0.0801	0.0402	0.0489	0.0637	0.0867	0.0469	0.0539	0.0315	0.0479	0.0225	0.0276
R^2Rec,-img†	**0.0559***	**0.0806***	**0.0412***	**0.0490***	**0.0641***	**0.0875***	**0.0474***	**0.0559***	**0.0327***	**0.0491***	**0.0229***	**0.0287***
▲%	4.49%	6.90%	3.52%	7.46%	5.78%	7.76%	5.57%	11.35%	2.19%	9.60%	2.69%	9.13%

text-based[17], image-based, and image-based cross-domain transfer learning[18], three R^2Rec, variants to fully verify the effectiveness of our proposed method.

7.3.3 Overall Performance (RQ1)

Overall Performance. The overall comparison results of our R^2Rec against other baselines are shown in Table 7.1. Based on the results, we could make the following observations: (1) ID-based solutions can also acquire competitive results; (2) text and image information can further improve the model performance; (3) tail-item-centered models can acquire comparable performance across all the solutions; (4) Our R^2Rec variant model, i.e., incorporating cross-domain transfer learning, achieves the best results on all three datasets, demonstrating the effectiveness of the proposed method.

Tail Item Performance. Figure 7.3 depicts the tail item performance on three datasets. Compared to the ID-based model, the tail item performance shows varying degrees of

[17]We extract item titles with a maximum of 10 tokens as the corresponding text information and apply the CLIP text encoder for item embedding initialization.

[18]We first pre-train the backbone on *Amazon Pantry*, *Amazon Clothing*, and *Amazon Magazine* datasets. Then, fine-tune it on the target domain.

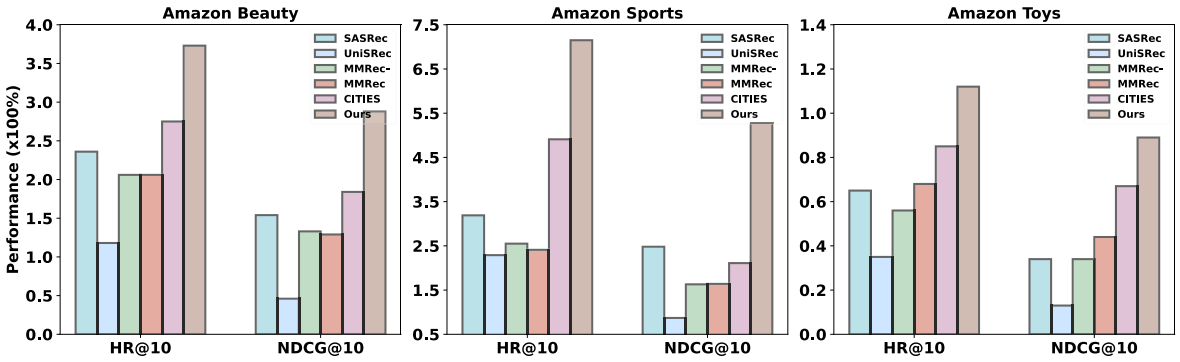


Figure 7.3: Tail item performance. R^2Rec, superiors to other baselines by a large margin.

Table 7.2: Item performance on the three datasets. The best results are highlighted in boldface, and the second-best results are underlined. "−" means the text modeling module is removed from raw models and only uses image information for recommendation. "w/o" and "w." mean with and without the specific module. † means cross-domain transfer learning is applied.

Dataset	Amazon Beauty				Amazon Toys				Amazon Sports			
	All		Tail		All		Tail		All		Tail	
	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10
R^2Rec,†	**0.0806**	**0.0490**	**0.0399**	**0.0307**	**0.0875**	**0.0559**	0.0696	0.0518	**0.0491**	**0.0287**	**0.0141**	**0.0112**
R^2Rec,	0.0801	0.0489	0.0373	0.0288	0.0867	0.0539	**0.0715**	**0.0528**	0.0479	0.0276	0.0138	0.0089
w/o reweight	0.0765	0.0475	0.0194	0.0129	0.0853	0.0541	0.0393	0.0251	0.0486	0.0286	0.0020	0.0012
w/o reembed	0.0728	0.0443	0.0302	0.0192	0.0835	0.0524	0.0568	0.0471	0.0435	0.0251	0.0075	0.0048
w/o R2	0.0762	0.0472	0.0078	0.0060	0.0857	0.0538	0.0108	0.0061	0.0475	0.0276	0.0003	0.0003
SASRec w. R2	0.0693	0.0445	0.0371	0.0299	0.0733	0.0489	0.0568	0.0471	0.0240	0.0157	0.0128	0.0107
MM-Rec$^-$ w. R2	0.0589	0.0411	0.0291	0.0183	0.0779	0.0452	0.0574	0.0439	0.0373	0.0197	0.0063	0.0059
MMMLP$^-$ w. R2	0.0516	0.0386	0.0255	0.0194	0.0530	0.0491	0.0403	0.0326	0.0317	0.0221	0.0043	0.0029
MM-Rec w. R2	0.0549	0.0364	0.0310	0.0213	0.0806	0.0478	0.0595	0.0457	0.0415	0.0269	0.0087	0.0083
MMMLP w. R2	0.0764	0.0457	0.0336	0.0233	0.0831	0.0539	0.0559	0.0481	0.0461	0.0267	0.0089	0.0068
CITIES w. R2	0.0663	0.0401	0.0189	0.0254	0.0718	0.0457	0.0613	0.0512	0.0368	0.0210	0.0121	0.0084
MAN w. R2	0.0724	0.0484	0.0388	**0.0315**	0.0730	0.0488	0.0574	0.0471	0.0459	0.0251	0.0095	0.0079

decline when incorporating text and image representation extracted by LLMs and LVMs. Our method outperforms tail-item-based solutions by a large margin, fully demonstrating the superiority of reweighting and reembedding strategies.

7.3.4 Ablation Studies (RQ2)

To fully demonstrate the effectiveness of reweighting and reembedding operations, we remove them from R^2Rec incrementally and also incorporate them into other baselines, observing the performance variation.

- **w/o Reweight.** Removing reweighting operation from R^2Rec,.

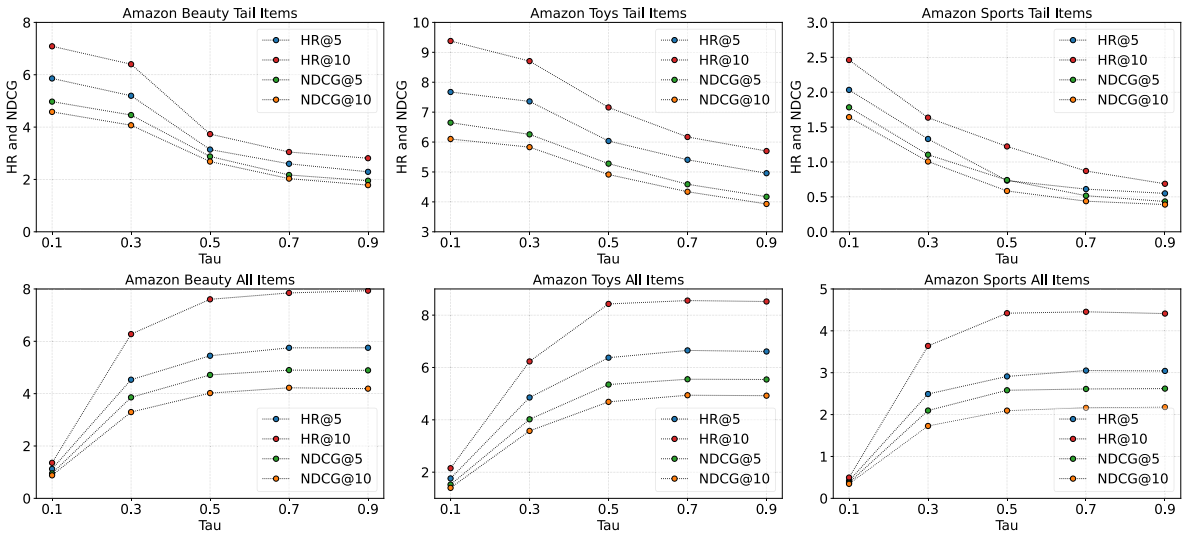


Figure 7.4: The performance with varying τ.

- **w/o Reembed.** Removing reembedding operation from R^{2}Rec,.
- **w R2.** Considering both reweighting and reembedding operations on other baselines[19].

Table 7.2 shows the ablation results. Note that removing either reembedding or reweighting operations leads to a measurable decline in the performance of all items, with a particularly notable impact on tail items. As the reweighting strategy is dedicated to facilitating the optimization of tail items, removing the reweighting function from R^{2}Rec results in a significant decline in the performance of tail items compared to the overall performance. Besides, from Table 7.1 and Table 7.2, we could find that these two operations also improve the other baselines' tail item performance, though the overall performance improvement is limited.

7.3.5 Impact of Hyperparameter (RQ3)

We investigate the impact of hyperparameter τ in Eq. (7.9) on the final performance. Mathematically, τ controls the shape of the weight distribution and further decides the attention assignment to the samples. As the hyperparameter τ decreases, the weight distribution converges toward a point mass, thereby allocating more attention to the tail

[19]Since some baselines (e.g., UniSRec, MELT, etc) do not utilize CE loss for model optimization, we exclude them from this experiment.

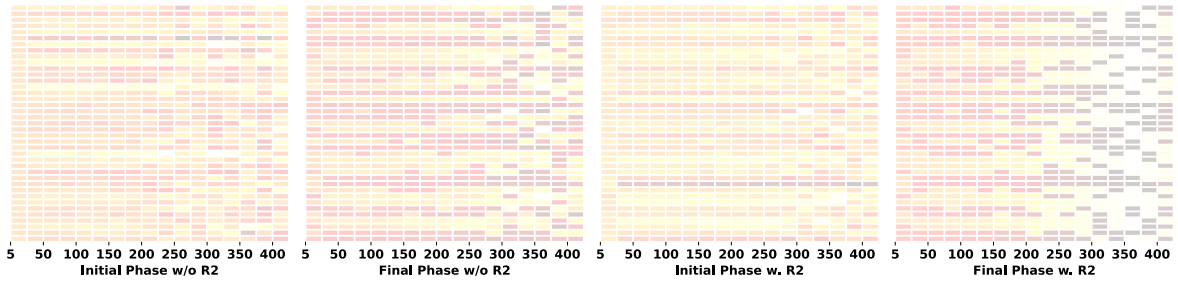


Figure 7.5: Visualization of item embedding with different interaction counts (from 5 to 400) at the initial and final stages of model training. The model w. R2 (i.e., with reembedding and reweighting strategies) acquires a sharper distribution on tail item (the item with fewer interactions) embeddings against the model w/o R2 (i.e., without reembedding and reweighting operations) after training.

items. Conversely, a larger value of τ will lead to a more uniform weight distribution, ensuring that each item will be treated equivalently. Due to insufficient training on tail items in comparison to popular items, the performance of tail items suffers from significant performance degeneration. Observing Figure 7.4, we could draw the consistent conclusion that as τ decreases, the tail items' performance can be improved while the overall performance will undergo a substantial decline. Overall, there is a trade-off between the performance on tail items and overall performance, a moderate τ (such as 0.5) is recommended to maintain a balance between them.

7.3.6 Embedding Visualization (RQ4)

To further explore the merits of reembedding and reweighting methods in tail item sequential recommendation, we examine the impact of removing or incorporating these operations on item embedding evolution. Specifically, we present the average item embeddings from the *Amazon Beauty* dataset, spanning interaction counts from 5 (i.e., tail items) to 400 (head items), observing the embedding variation during the model training process, as shown in Figure 7.5. We can observe that in the initial phase of the model training, the item embeddings exhibit a uniform distribution for both models with or without reembedding and reweighting operations. As training progresses, the head items (items with numerous interactions) and tail items (items with fewer interactions) in our R^2Rec, are optimized sufficiently, resulting in a more sharply concentrated embedding distribution. In contrast, for the model without remembedding and reweighting operations, the tail item embeddings demonstrate minimal variation after training. Con-

sequently, we argue that the proposed two operations can enhance the model's learning ability in tail items.

7.4 Conclusion

This work attempts to alleviate the tail item performance degeneration on image-based sequential recommendations. To instantiate this idea, we first analyze the deficiency of standard CE loss and image-based or text-based embeddings, respectively, then propose *all in ground-truth* and *knowledge transfer tax* two perspectives contributing to this problem. From these two considerations, we further propose reweighting and reembedding functions for recommendation, named R^2Rec. Specifically, instead of treating the head items and tail items equally and focusing solely on the ground-truth, the reweighting strategy allows the model to adaptively assign more attention to tail items during the model optimization, alleviating the insufficient training of tail items. Reembedding operation initializes the tail item embedding via a standard Gaussian distribution, tackling the negative transfer of external knowledge encapsulated in LLMs and LVMs. Theoretically, our reweighting function is similar to DPO in LLMs preference alignment, but could achieve a more precise optimization. Comprehensive experiments on three public datasets manifest that our R^2Rec is superior to the baselines on overall performance and tail items. Furthermore, when integrated with other baselines, our method can achieve additional performance improvements.

CHAPTER 8

CONCLUSION

In this dissertation, we gave readers a thorough overview of graph neural networks for loan default risk prediction and recommendation, as well as how we contributed to the development of these two tasks since they are significant in practical scenarios.

In Chapter 2, we comprehensively review the development of representative solutions for loan default risk prediction. More concertedly, early attempts heavily rely on professional domain knowledge and expert experience for rule design (e.g., '5C' Principles) and risk application identification. This process is knowledge-driven and labor-intensive. Facing the prosperity of loan application requirements and services, semi-automatic solutions based on machine learning models are proposed, including decision tree, Random Forest, and XGBoost. Owing to the good explainability and low computational costs, they achieved promising results and are still prominent schemes in practical applications. Deep learning methods emerged and became ubiquitous in a wide range of tasks since breaking the boundaries of AlexNet in 2012. Using deep learning models for loan default prediction attracted widespread attention from academics and the industry. The two mainstream recipes can be categorized into: deep neural networks (e.g., attention-based and MLP) and graph neural networks. Given the rapid developments of deep learning, many cutting-edge models are also adopted for this task.

In Chapter 3, we mainly focus on the issue of missing values in raw loan applications, which hinders the prediction accuracy from being further improved. To overcome this issue, we propose a loan application graph where similar application records could be connected based on the graph structures, thus, the auxiliary information from neighbors

or similar records can be integrated for current application prediction. Experimental results illustrate the superiority of our graph-based models in loan default risk detection against conventional machine learning methods and other deep neural networks.

In Chapter 4, we further move the lens to the highly unbalanced application records distribution, i.e., the default records only account for a very small part of all the data. The skewed data distribution results in the low recall value of predictions. To address this problem, we designed multi-view loan application graphs. Specifically, by adjusting the similarity threshold between two applications, we could attain a series of graph structures. Therefore, a set of representations can be yielded based on graph neural networks. Afterward, the multi-view representations can be leveraged for small data enhancement and prediction. The experiments demonstrate that our multi-view graph enhancement strategy outperforms the popular sample-based solutions by a large margin.

As for recommendation, we explore the graph neural networks in recommendation systems. Specifically, in Chapter 5, we first walked through the history of session-based recommendation, which dates back to the 2000s, when the session concept was first proposed in recommendation systems. Different from other recommendation tasks, session-based recommendation emphasizes short-term preference modeling and dynamic recommendation in the current session. Reviewing existing methods, we organize them into two categories: sequential neural networks and graph neural networks, both of which are the main solutions for session-based recommendation. More concretely, we discuss the characteristics of session-based recommendation from four aspects, including the limited session length, the timeliness of recommendations, the independence of orders, and the anonymity of user information. Various graph structures in session-based recommendation are also introduced. Finally, we conclude the challenges and future development directions in this research avenue.

In Chapter 6, we argue that the implicit relationships (i.e., co-occurrence relationships in interaction records) between items are equally important compared with explicit relationships (i.e., adjacent relationships in interaction records). However, this information is chronically overlooked by most graph-based methods. To model this type of connection, we proposed an internal graph neural network, which learns the global connections between items adaptively and integrates with the external graph for item representation fusion and recommendation. The in-depth analysis illustrates the merits of our dual graph neural networks in alignment and uniformity, the two properties.

In Chapter 7, we show that the external knowledge derived from language models or vision models heavily hurts the performance of tail-item recommendation against

ID-based methods, albeit the overall performance can be improved. Therefore, we propose two potential inducements: all-in targets (i.e., the model focuses solely on optimizing the target items, while considering all other items as equally negative, which is inconsistent with real-world scenarios.); and knowledge transfer tax (i.e., as the tail item can not acquire fully attention and optimization, the external knowledge encapsulated in the language and vision models steer the directions of model optimization, which is inappropriate as the recommendation relies more on the collaborate knowledge instead of the open-world knowledge). Consequently, to address these two challenges, we introduce two corresponding solutions, i.e., re-embedding and re-weighting. The first solution involves re-initializing the tail-item embeddings with a standard normal distribution in order to alleviate the negative knowledge transfer arising from the language and vision models. The re-weighting method allows us to allocate attention to different items dynamically, guiding the model to pay more attention to the tail items when optimizing to alleviate the all-in target issue. Extensive experiments present the effectiveness of our solution. Moreover, these two modules can be integrated into other backbones effectively to improve the tail-item performance.

To sum up, in this thesis, we first employ graph neural networks for credit risk prediction. Besides, a multi-view graph structure is proposed to tackle missing values and data imbalance in loan default risk prediction. We provide a comprehensive review of over 150 studies on graph neural networks for session-based recommendation. Inspired by the existing studies, we propose internal and external graph neural networks for session-based recommendation. Moreover, we introduce an efficient reembedding and resampling strategy to mitigate the tail item recommendation problem, which can be flexibly integrated into various base models to enhance performance. We are thrilled with the progress made in this field over the past three years. At the same time, significant challenges remain and deserve to be addressed in the future. One key challenge is how to efficiently update our deep neural networks online and how to capture and model the user's ongoing preference evolution effectively for a real-time recommendation. In the future, we will cultivate what is being discussed, rather than just modeling from static small-scale data, to reach the next level in the recommendation system.

We believe that our proposed graph neural network architectures, along with the strategies for addressing data imbalance, hold significant potential for practical applications across diverse domains, including but not limited to finance, healthcare, and transportation. We hope our work inspires other researchers to explore these issues or to extend our methods to new tasks and domains. Together, these efforts can help create

more effective recommendation systems and foster their meaningful integration into industrial applications.

BIBLIOGRAPHY

[1] C. R. ABRAHAMS AND M. ZHANG, *Fair lending compliance: Intelligence and implications for credit risk management*, (2008).

[2] P. M. ADDO, D. GUEGAN, AND B. HASSANI, *Credit risk analysis using machine and deep learning models*, Risks, 6 (2018), p. 38.

[3] G. ADOMAVICIUS AND A. TUZHILIN, *Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions*, IEEE transactions on knowledge and data engineering, 17 (2005), pp. 734–749.

[4] Y. ALEKSANDROVA, *Comparing performance of machine learning algorithms for default risk prediction in peer to peer lending*, TEM Journal, 10 (2021), pp. 133–143.

[5] D. BABAEV, M. SAVCHENKO, A. TUZHILIN, AND D. UMERENKOV, *Et-rnn: Applying deep learning to credit loan applications*, in Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, 2019, pp. 2183–2190.

[6] K. BAO, J. ZHANG, Y. ZHANG, W. WANG, F. FENG, AND X. HE, *Tallrec: An effective and efficient tuning framework to align large language model with recommendation*, in Proceedings of the 17th ACM Conference on Recommender Systems, 2023, pp. 1007–1014.

[7] T. BELLOTTI AND J. CROOK, *Forecasting and stress testing credit card default using dynamic models*, International Journal of Forecasting, 29 (2013), pp. 563–574.

[8] S. BHATTACHARYYA, S. JHA, K. THARAKUNNEL, AND J. C. WESTLAND, *Data mining for credit card fraud: A comparative study*, Decision support systems, 50 (2011), pp. 602–613.

[9] S. BIAN, X. PAN, W. X. ZHAO, J. WANG, C. WANG, AND J.-R. WEN, *Multi-modal mixture of experts represetation learning for sequential recommendation*, in Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, 2023, pp. 110–119.

[10] A. BOJCHEVSKI AND S. GÜNNEMANN, *Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking*, arXiv preprint arXiv:1707.03815, (2017).

[11] A. BORDES, N. USUNIER, A. GARCIA-DURAN, J. WESTON, AND O. YAKHNENKO, *Translating embeddings for modeling multi-relational data*, Advances in neural information processing systems, 26 (2013).

[12] L. BOURTOULE, V. CHANDRASEKARAN, C. A. CHOQUETTE-CHOO, H. JIA, A. TRAVERS, B. ZHANG, D. LIE, AND N. PAPERNOT, *Machine unlearning*, in Proc. of SP, 2021, pp. 141–159.

[13] G. E. BOX AND R. D. MEYER, *An analysis for unreplicated fractional factorials*, Technometrics, 28 (1986), pp. 11–18.

[14] R. A. BRADLEY AND M. E. TERRY, *Rank analysis of incomplete block designs: I. the method of paired comparisons*, Biometrika, 39 (1952), pp. 324–345.

[15] L. BREIMAN, *Random forests*, Machine Learning archive, 45 (2001), pp. 5–32.

[16] D. CARRARO AND D. BRIDGE, *Enhancing recommendation diversity by re-ranking with large language models*, arXiv preprint arXiv:2401.11506, (2024).

[17] P. CASTELLS, N. HURLEY, AND S. VARGAS, *Novelty and diversity in recommender systems*, in Recommender systems handbook, 2021, pp. 603–646.

[18] N. V. CHAWLA, K. W. BOWYER, L. O. HALL, AND W. P. KEGELMEYER, *Smote: synthetic minority over-sampling technique*, Journal of artificial intelligence research, 16 (2002), pp. 321–357.

[19] C. CHEN, J. GUO, AND B. SONG, *Dual attention transfer in session-based recommendation with multi-dimensional integration*, in Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 869–878.

[20] H. CHEN, X. CHEN, S. SHI, AND Y. ZHANG, *Generate natural language explanations for recommendation*, arXiv preprint arXiv:2101.03392, (2021).

[21] 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, (2023), pp. 1–39.

[22] J. CHEN, H. LI, X. ZHANG, F. ZHANG, S. WANG, K. WEI, AND J. JI, *Sr-hetgnn: session-based recommendation with heterogeneous graph neural network*, Knowl. Inf. Syst., 66 (2023), p. 1111,Äì1134.

[23] M. CHEN, Z. WEI, Z. HUANG, B. DING, AND Y. LI, *Simple and deep graph convolutional networks*, in International Conference on Machine Learning, PMLR, 2020, pp. 1725–1735.

[24] M. CHEN AND J. ZHENG, *Incorporating adjacent user modeling into session-based recommendation with graph neural networksstreaming session-based recommendation*, in 2021 International Conference on Data Mining Workshops (ICDMW), IEEE, 2021, pp. 1–9.

[25] N. CHEN, B. RIBEIRO, AND A. CHEN, *Financial credit risk assessment: a recent review*, Artificial Intelligence Review, 45 (2016), pp. 1–23.

[26] Q. CHEN, Z. GUO, J. LI, AND G. LI, *Knowledge-enhanced multi-view graph neural networks for session-based recommendation*, in Proc. of SIGIR, 2023.

[27] Q. CHEN, J. LI, Z. GUO, G. LI, AND Z. DENG, *Attribute-enhanced dual channel representation learning for session-based recommendation*, in Proc. of CIKM, 2023, pp. 3793–3797.

[28] T. CHEN AND C. GUESTRIN, *Xgboost: A scalable tree boosting system*, in 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 785–794.

[29] T. CHEN AND R. C.-W. WONG, *Session-based recommendation with local invariance*, in 2019 IEEE International Conference on Data Mining (ICDM), IEEE, 2019, pp. 994–999.

[30] T. CHEN AND R. C. W. WONG, *Handling information loss of graph neural networks for session-based recommendation*, in Proceedings of the 26th ACM SIGKDD

International Conference on Knowledge Discovery and Data Mining, 2020, pp. 1172–1180.

[31] ——, *An efficient and effective framework for session-based social recommendation*, in Proceedings of the 14th ACM International Conference on Web Search and Data Mining, 2021, pp. 400–408.

[32] W. CHEN, F. CAI, H. CHEN, AND M. DE RIJKE, *A dynamic co-attention network for session-based recommendation*, in Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 1461–1470.

[33] X. CHEN, H. CHEN, H. XU, Y. ZHANG, Y. CAO, Z. QIN, AND H. ZHA, *Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation*, in Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 765–774.

[34] X. CHEN, H. XU, Y. ZHANG, J. TANG, Y. CAO, Z. QIN, AND H. ZHA, *Sequential recommendation with user memory networks*, in Proc. of WSDM, 2018.

[35] X. CHEN, L. YAO, J. MCAULEY, G. ZHOU, AND X. WANG, *Deep reinforcement learning in recommender systems: A survey and new perspectives*, Knowledge-Based Systems, 264 (2023), p. 110335.

[36] Y. CHEN, Z. LI, C. YANG, X. WANG, G. LONG, AND G. XU, *Adaptive graph recurrent network for multivariate time series imputation*, in Proc. of ICONIP, Springer, 2022, pp. 64–73.

[37] Y. CHEN, K. SHI, Z. WU, J. CHEN, X. WANG, J. MCAULEY, G. XU, AND S. YU, *Temporal disentangled contrastive diffusion model for spatiotemporal imputation*, arXiv preprint arXiv:2402.11558, (2024).

[38] H.-T. CHENG, L. KOC, J. HARMSEN, T. SHAKED, T. CHANDRA, H. ARADHYE, G. ANDERSON, G. CORRADO, W. CHAI, M. ISPIR, ET AL., *Wide and deep learning for recommender systems*, in 1st workshop on deep learning for recommender systems, 2016, pp. 7–10.

[39] B.-W. CHI AND C.-C. HSU, *A hybrid approach to integrate genetic algorithm into dual scoring model in enhancing the performance of credit scoring model*, Expert systems with applications, 39 (2012), pp. 2650–2661.

[40] J. CHI, G. ZENG, Q. ZHONG, T. LIANG, J. FENG, X. AO, AND J. TANG, *Learning to undersampling for class imbalanced credit risk forecasting*, in 2020 IEEE International Conference on Data Mining (ICDM), IEEE, 2020, pp. 72–81.

[41] M. CHOI, J. KIM, J. LEE, H. SHIM, AND J. LEE, *S-walk: Accurate and scalable session-based recommendation with random walks*, in Proc. of WSDM, 2022, pp. 150–160.

[42] C. L. CLARKE, M. KOLLA, G. V. CORMACK, O. VECHTOMOVA, A. ASHKAN, S. BÜTTCHER, AND I. MACKINNON, *Novelty and diversity in information retrieval evaluation*, in Proc. of SIGIR, 2008, pp. 659–666.

[43] E. CONTAL AND G. MCGOLDRICK, *Ragsys: Item-cold-start recommender as rag system*, arXiv preprint arXiv:2405.17587, (2024).

[44] A. COSER, M. M. MAER-MATEI, AND C. ALBU, *Predictive models for loan default risk assessment*, Economic Computation and Economic Cybernetics Studies and Research, 53 (2019), pp. 149–165.

[45] D. R. COX, *The regression analysis of binary sequences*, Journal of the Royal Statistical Society Series B: Statistical Methodology, 20 (1958), pp. 215–232.

[46] F.-A. CROITORU, V. HONDRU, R. T. IONESCU, AND M. SHAH, *Diffusion models in vision: A survey*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 45 (2023), pp. 10850–10869.

[47] J. N. CROOK, D. B. EDELMAN, AND L. C. THOMAS, *Recent developments in consumer credit risk assessment*, European Journal of Operational Research, 183 (2007), pp. 1447–1465.

[48] L. CUI, L. BAI, Y. WANG, X. BAI, Z. ZHANG, AND E. R. HANCOCK, *P2p lending analysis using the most relevant graph-based features*, in Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR), Springer, 2016, pp. 3–14.

[49] Z. CUI, H. CHEN, L. CUI, S. LIU, X. LIU, G. XU, AND H. YIN, *Reinforced kgs reasoning for explainable sequential recommendation*, World Wide Web, 25 (2022), pp. 631–654.

[50] Z. CUI, J. MA, C. ZHOU, J. ZHOU, AND H. YANG, *M6-rec: Generative pre-trained language models are open-ended recommender systems*, arXiv preprint arXiv:2205.08084, (2022).

[51] J. DAVIDSON, B. LIEBALD, J. LIU, P. NANDY, T. VAN VLEET, U. GARGI, S. GUPTA, Y. HE, M. LAMBERT, B. LIVINGSTON, AND D. SAMPATH, *The youtube video recommendation system*, in Proceedings of the Fourth ACM Conference on Recommender Systems, RecSys '10, New York, NY, USA, 2010, Association for Computing Machinery, p. 293,Äì296.

[52] G. DE SOUZA PEREIRA MOREIRA, S. RABHI, J. M. LEE, R. AK, AND E. OLDRIDGE, *Transformers4rec: Bridging the gap between nlp and sequential / session-based recommendation*, in Fifteenth ACM Conference on Recommender Systems, 2021, pp. 143–153.

[53] Y. DELDJOO, Z. HE, J. MCAULEY, A. KORIKOV, S. SANNER, A. RAMISA, R. VIDAL, M. SATHIAMOORTHY, A. KASIRZADEH, AND S. MILANO, *A review of modern recommender systems using generative models (gen-recsys)*, in Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD '24, New York, NY, USA, 2024, Association for Computing Machinery, p. 6448,Äì6458.

[54] A. DENG AND B. HOOI, *Graph neural network-based anomaly detection in multivariate time series*, in Proc. of AAAI, 2021.

[55] Z.-H. DENG, C.-D. WANG, L. HUANG, J.-H. LAI, AND S. Y. PHILIP, *G 3 sr: Global graph guided session-based recommendation*, IEEE transactions on neural networks and learning systems, 34 (2022), pp. 9671–9684.

[56] T. EBESU, B. SHEN, AND Y. FANG, *Collaborative memory network for recommendation systems*, in Proc. of SIGIR, 2018.

[57] A. EDMUNDS AND A. MORRIS, *The problem of information overload in business organisations: a review of the literature*, International journal of information management, 20 (2000), pp. 17–28.

[58] E. ELAHI, S. ANWAR, B. SHAH, Z. HALIM, A. ULLAH, I. RIDA, AND M. WAQAS, *Knowledge graph enhanced contextualized attention-based network for responsible user-specific recommendation*, ACM Trans. Intell. Syst. Technol., 15 (2024).

[59] Z. FAN, Z. LIU, S. WANG, L. ZHENG, AND P. S. YU, *Modeling sequences as distributions with uncertainty for sequential recommendation*, in Proc. of CIKM, 2021.

[60] Z. FAN, Z. LIU, Y. WANG, A. WANG, Z. NAZARI, L. ZHENG, H. PENG, AND P. S. YU, *Sequential recommendation via stochastic self-attention*, in Proceedings of the ACM web conference 2022, 2022, pp. 2036–2047.

[61] H. FANG, D. ZHANG, Y. SHU, AND G. GUO, *Deep learning for sequential recommendation: Algorithms, influential factors, and evaluations*, ACM Transactions on Information Systems (TOIS), 39 (2020), pp. 1–42.

[62] J. H. FRIEDMAN, *Greedy function approximation: a gradient boosting machine*, Annals of statistics, (2001), pp. 1189–1232.

[63] Z. FU, Y. XIAN, R. GAO, J. ZHAO, Q. HUANG, Y. GE, S. XU, S. GENG, C. SHAH, Y. ZHANG, ET AL., *Fairness-aware explainable recommendation over knowledge graphs*, in Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 69–78.

[64] Y. GAO, Y. XIONG, X. GAO, K. JIA, J. PAN, Y. BI, Y. DAI, J. SUN, AND H. WANG, *Retrieval-augmented generation for large language models: A survey*, arXiv preprint arXiv:2312.10997, (2023).

[65] S. GENG, S. LIU, Z. FU, Y. GE, AND Y. ZHANG, *Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5)*, in Proceedings of the 16th ACM Conference on Recommender Systems, 2022, pp. 299–315.

[66] S. GENG, J. TAN, S. LIU, Z. FU, AND Y. ZHANG, *Vip5: Towards multimodal foundation models for recommendation*, in Findings of the Association for Computational Linguistics: EMNLP 2023, 2023, pp. 9606–9620.

[67] A. GROVER AND J. LESKOVEC, *node2vec: Scalable feature learning for networks*, in Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 2016, pp. 855–864.

[68] C. GUO, M. ZHANG, J. FANG, J. JIN, AND M. PAN, *Session-based recommendation with hierarchical leaping networks*, in Proceedings of the 43rd International

ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 1705–1708.

[69] J. GUO, Y. YANG, X. SONG, Y. ZHANG, Y. WANG, J. BAI, AND Y. ZHANG, *Learning multi-granularity consecutive user intent unit for session-based recommendation*, in Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, 2022, pp. 343–352.

[70] L. GUO, H. YIN, Q. WANG, T. CHEN, A. ZHOU, AND N. QUOC VIET HUNG, *Streaming session-based recommendation*, in Proc. of KDD, 2019.

[71] X. GUO, Y. QUAN, H. ZHAO, Q. YAO, Y. LI, AND W. TU, *Tabgnn: Multiplex graph neural network for tabular data prediction*, arXiv preprint arXiv:2108.09127, (2021).

[72] P. HAJEK AND K. MICHALAK, *Feature selection in corporate credit rating prediction*, Knowledge-Based Systems, 51 (2013), pp. 72–84.

[73] W. L. HAMILTON, R. YING, AND J. LESKOVEC, *Inductive representation learning on large graphs*, in Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, Red Hook, NY, USA, 2017, Curran Associates Inc., p. 1025,Äì1035.

[74] D. J. HAND AND W. E. HENLEY, *Statistical classification methods in consumer credit scoring: a review*, Journal of the Royal Statistical Society: Series A (Statistics in Society), 160 (1997), pp. 523–541.

[75] D. J. HAND AND W. E. HENLEY, *Statistical classification methods in consumer credit scoring: A review*, Journal of the Royal Statistical Society. Series A (Statistics in Society), 160 (1997), pp. 523–541.

[76] J. HARTE, W. ZORGDRAGER, P. LOURIDAS, A. KATSIFODIMOS, D. JANNACH, AND M. FRAGKOULIS, *Leveraging large language models for sequential recommendation*, in Proceedings of the 17th ACM Conference on Recommender Systems, 2023, pp. 1096–1102.

[77] S. HE, K. LIU, G. JI, AND J. ZHAO, *Learning to represent knowledge graphs with gaussian embedding*, in Proceedings of the 24th ACM international on conference on information and knowledge management, 2015.

[78] X. HE, K. DENG, X. WANG, Y. LI, Y. ZHANG, AND M. WANG, *Lightgcn: Simplifying and powering graph convolution network for recommendation*, in Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, 2020, pp. 639–648.

[79] O. C. HERRADA, *Music recommendation and discovery in the long tail*, Universitat Pompeu Fabra, 2009.

[80] B. HIDASI AND A. KARATZOGLOU, *Recurrent neural networks with top-k gains for session-based recommendations*, in Proceedings of the 27th ACM international conference on information and knowledge management, 2018, pp. 843–852.

[81] B. HIDASI, A. KARATZOGLOU, L. BALTRUNAS, AND D. TIKK, *Session-based recommendations with recurrent neural networks*, arXiv preprint arXiv:1511.06939, (2015).

[82] G. E. HINTON AND R. R. SALAKHUTDINOV, *Reducing the dimensionality of data with neural networks*, science, 313 (2006), pp. 504–507.

[83] J. HO, A. JAIN, AND P. ABBEEL, *Denoising diffusion probabilistic models*, Proc. of NeurIPS, (2020), pp. 6840–6851.

[84] W. HOEFFDING, *Probability inequalities for sums of bounded random variables*, The collected works of Wassily Hoeffding, (1994), pp. 409–426.

[85] J. HONG, N. LEE, AND J. THORNE, *Orpo: Monolithic preference optimization without reference model*, arXiv preprint arXiv:2403.07691, 2 (2024), p. 5.

[86] Y. HOU, B. HU, Z. ZHANG, AND W. X. ZHAO, *Core: simple and effective session-based recommendation within consistent representation space*, in Proc. of SIGIR, 2022, pp. 1796–1801.

[87] Y. HOU, S. MU, W. X. ZHAO, Y. LI, B. DING, AND J.-R. WEN, *Towards universal sequence representation learning for recommender systems*, in Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2022, pp. 585–593.

[88] Y. HOU, J. ZHANG, Z. LIN, H. LU, R. XIE, J. MCAULEY, AND W. X. ZHAO, *Large language models are zero-shot rankers for recommender systems*, in European Conference on Information Retrieval, 2024, pp. 364–381.

[89] B. HU, Z. ZHANG, J. ZHOU, J. FANG, Q. JIA, Y. FANG, Q. YU, AND Y. QI, *Loan default analysis with multiplex graph learning*, in Proceedings of the 29th ACM international conference on information & knowledge management, 2020, pp. 2525–2532.

[90] E. J. HU, Y. SHEN, P. WALLIS, Z. ALLEN-ZHU, Y. LI, S. WANG, L. WANG, AND W. CHEN, *Lora: Low-rank adaptation of large language models*, arXiv preprint arXiv:2106.09685, (2021).

[91] L. HU, L. CAO, S. WANG, G. XU, J. CAO, AND Z. GU, *Diversifying personalized recommendation with user-session context.*, in Proc. of IJCAI, 2017.

[92] Y. HU, Y. LIU, C. MIAO, AND Y. MIAO, *Memory bank augmented long-tail sequential recommendation*, in Proceedings of the 31st ACM International Conference on Information & Knowledge Management, 2022, pp. 791–801.

[93] C. HUANG, J. CHEN, L. XIA, Y. XU, P. DAI, Y. CHEN, L. BO, J. ZHAO, AND J. X. HUANG, *Graph-enhanced multi-task learning of multi-level transition dynamics for session-based recommendation*, in AAAI Conference on Artificial Intelligence (AAAI), 2021.

[94] C.-L. HUANG, M.-C. CHEN, AND C.-J. WANG, *Credit scoring with a data mining approach based on support vector machines*, Expert systems with applications, 33 (2007), pp. 847–856.

[95] J.-J. HUANG, G.-H. TZENG, AND C.-S. ONG, *Two-stage genetic programming (2sgp) for the credit scoring model*, Applied Mathematics and Computation, 174 (2006), pp. 1039–1053.

[96] Z. HUANG, H. CHEN, C.-J. HSU, W.-H. CHEN, AND S. WU, *Credit rating analysis with support vector machines and neural networks: a market comparative study*, Decision support systems, 37 (2004), pp. 543–558.

[97] A. JAGATAP, N. GUPTA, S. FARFADE, AND P. M. COMAR, *Attribert: Session-based product attribute recommendation with bert*, (2023).

[98] S. JANG, H. LEE, H. CHO, AND S. CHUNG, *Cities: Contextual inference of tail-item embeddings for sequential recommendation*, in 2020 IEEE International Conference on Data Mining (ICDM), IEEE, 2020, pp. 202–211.

[99] D. JANNACH AND M. LUDEWIG, *When recurrent neural networks meet the neighborhood for session-based recommendation*, in Proceedings of the eleventh ACM conference on recommender systems, 2017, pp. 306–310.

[100] B. JIA, J. CAO, S. QIAN, N. ZHU, X. DONG, L. ZHANG, L. CHENG, AND L. MO, *Smone: A session-based recommendation model based on neighbor sessions with similar probabilistic intentions*, ACM Transactions on Knowledge Discovery from Data, 17 (2023), pp. 1–22.

[101] M. JIANG, Y. ZHANG, Y. GAO, Y. WANG, F. FENG, AND X. HE, *Lightmirm: Light meta-learned invariant risk minimization for trustworthy loan default prediction*, in 2023 IEEE 39th International Conference on Data Engineering (ICDE), IEEE, 2023, pp. 3494–3507.

[102] W. JIANG AND J. LUO, *Graph neural network for traffic forecasting: A survey*, Expert systems with applications, (2022), p. 117921.

[103] W.-C. KANG AND J. MCAULEY, *Self-attentive sequential recommendation*, in ICDM, IEEE, 2018, pp. 197–206.

[104] G. KE, Q. MENG, T. FINLEY, T. WANG, W. CHEN, W. MA, Q. YE, AND T.-Y. LIU, *Lightgbm: A highly efficient gradient boosting decision tree*, Advances in neural information processing systems, 30 (2017), pp. 3146–3154.

[105] H. S. KIM AND S. Y. SOHN, *Support vector machines for default prediction of smes based on technology credit*, European Journal of Operational Research, 201 (2010), pp. 838–846.

[106] K. KIM, D. HYUN, S. YUN, AND C. PARK, *Melt: Mutual enhancement of long-tailed user and item for sequential recommendation*, in Proceedings of the 46th international ACM SIGIR conference on Research and development in information retrieval, 2023, pp. 68–77.

[107] T. N. KIPF AND M. WELLING, *Semi-supervised classification with graph convolutional networks*, arXiv preprint arXiv:1609.02907, (2016).

[108] B. KIUMARSI, K. G. VAMVOUDAKIS, H. MODARES, AND F. L. LEWIS, *Optimal and autonomous control using reinforcement learning: A survey*, IEEE transactions on neural networks and learning systems, 29 (2017), pp. 2042–2062.

[109] T. KOJIMA, S. S. GU, M. REID, Y. MATSUO, AND Y. IWASAWA, *Large language models are zero-shot reasoners*, Advances in neural information processing systems, 35 (2022), pp. 22199–22213.

[110] Y. KOREN, R. BELL, AND C. VOLINSKY, *Matrix factorization techniques for recommender systems*, Computer, 42 (2009), pp. 30–37.

[111] W. KRICHENE AND S. RENDLE, *On sampled metrics for item recommendation*, Communications of the ACM, 65 (2022), pp. 75–83.

[112] A. KRIZHEVSKY, I. SUTSKEVER, AND G. E. HINTON, *Imagenet classification with deep convolutional neural networks*, Advances in neural information processing systems, 25 (2012).

[113] H. KVAMME, N. SELLEREITE, K. AAS, AND S. SJURSEN, *Predicting mortgage default using convolutional neural networks*, Expert Systems with Applications, 102 (2018), pp. 207–217.

[114] S. LAI, E. MENG, F. ZHANG, C. LI, B. WANG, AND A. SUN, *An attribute-driven mirror graph network for session-based recommendation*, in Proc. of SIGIR, 2022.

[115] N. LATHIA, S. HAILES, AND L. CAPRA, *knn cf: a temporal social network*, in Proceedings of the 2008 ACM conference on Recommender systems, 2008, pp. 227–234.

[116] Y. LECUN, Y. BENGIO, AND G. HINTON, *Deep learning*, nature, 521 (2015), pp. 436–444.

[117] T.-S. LEE, C.-C. CHIU, C.-J. LU, AND I.-F. CHEN, *Credit scoring using the hybrid neural discriminant technique*, Expert Systems with applications, 23 (2002), pp. 245–254.

[118] A. LI, Z. CHENG, F. LIU, Z. GAO, W. GUAN, AND Y. PENG, *Disentangled graph neural networks for session-based recommendation*, IEEE Transactions on Knowledge and Data Engineering, 35 (2022), pp. 7870–7882.

[119] C. LI, C. QUAN, L. PENG, Y. QI, Y. DENG, AND L. WU, *A capsule network for recommendation and explaining what you like and dislike*, in Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, 2019, pp. 275–284.

[120] D. LI, D. CHEN, B. JIN, L. SHI, J. GOH, AND S.-K. NG, *Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks*, in International conference on artificial neural networks, Springer, 2019, pp. 703–716.

[121] G. LI, Y. SHI, AND Z. ZHANG, *P2p default risk prediction based on xgboost, svm and rf fusion model*, in 1st International Conference on Business, Economics, Management Science, 2019, pp. 470–475.

[122] J. LI, D. CAI, AND X. HE, *Learning graph-level representation for drug discovery*, arXiv preprint arXiv:1709.03741, (2017).

[123] J. LI, M. WANG, J. LI, J. FU, X. SHEN, J. SHANG, AND J. MCAULEY, *Text is all you need: Learning language representations for sequential recommendation*, arXiv preprint arXiv:2305.13731, (2023).

[124] L. LI, Y. ZHANG, AND L. CHEN, *Personalized transformer for explainable recommendation*, in Proc. of ACL, 2021, pp. 4947–4957.

[125] ——, *Personalized prompt learning for explainable recommendation*, ACM Transactions on Information Systems, 41 (2023), pp. 1–26.

[126] P. LI, Z. WANG, Z. REN, L. BING, AND W. LAM, *Neural rating regression with abstractive tips generation for recommendation*, in Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval, 2017, pp. 345–354.

[127] Q. LI, Z. HAN, AND X.-M. WU, *Deeper insights into graph convolutional networks for semi-supervised learning*, in Thirty-Second AAAI conference on artificial intelligence, 2018.

[128] X. LI, A. SUN, M. ZHAO, J. YU, K. ZHU, D. JIN, M. YU, AND R. YU, *Multi-intention oriented contrastive learning for sequential recommendation*, in Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, 2023, pp. 411–419.

[129] Y. LI, *Credit risk prediction based on machine learning methods*, in 14th International Conference on Computer Science & Education, IEEE, 2019, pp. 1011–1013.

[130] Y. LI, T. CHEN, P.-F. ZHANG, AND H. YIN, *Lightweight self-attentive sequential recommendation*, in Proc. of CIKM, 2021, pp. 967–977.

[131] Y. LI, C. GAO, X. DU, H. WEI, H. LUO, D. JIN, AND Y. LI, *Spatiotemporal-aware session-based recommendation with graph neural networks*, in Proc. of CIKM, 2022.

[132] Y. LI, C. GAO, H. LUO, D. JIN, AND Y. LI, *Enhancing hypergraph neural networks with intent disentanglement for session-based recommendation*, in Proc. of SIGIR, 2022.

[133] Y. LI, D. TARLOW, M. BROCKSCHMIDT, AND R. ZEMEL, *Gated graph sequence neural networks*, arXiv preprint arXiv:1511.05493, (2015).

[134] Z. LI, A. SUN, AND C. LI, *Diffurec: A diffusion model for sequential recommendation*, ACM Transactions on Information Systems, 42 (2023), pp. 1–28.

[135] Z. LI, R. WANG, K. CHEN, M. UTIYAMA, E. SUMITA, Z. ZHANG, AND H. ZHAO, *Data-dependent gaussian prior objective for language generation*, in International Conference on Learning Representations, 2020.

[136] Z. LI, X. WANG, C. YANG, L. YAO, J. MCAULEY, AND G. XU, *Exploiting explicit and implicit item relationships for session-based recommendation*, in Proc. of WSDM, 2023.

[137] Z. LI, X. WANG, L. YAO, Y. CHEN, G. XU, AND E.-P. LIM, *Graph neural network with self-attention and multi-task learning for credit default risk prediction*, in Proc. of WISE, Springer, 2022, pp. 616–629.

[138] Z. LI, Y. XIE, W. E. ZHANG, P. WANG, L. ZOU, F. LI, X. LUO, AND C. LI, *Disentangle interest trend and diversity for sequential recommendation*, Information Processing & Management, 61 (2024), p. 103619.

[139] D. LIANG, J. ALTOSAAR, L. CHARLIN, AND D. M. BLEI, *Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence*, in Proc. of RecSys, 2016.

[140] J. LIANG, X. ZHAO, M. LI, Z. ZHANG, W. WANG, H. LIU, AND Z. LIU, *Mmmlp: Multi-modal multilayer perceptron for sequential recommendations*, in WWW, 2023, pp. 1109–1117.

[141] G. LIN, C. GAO, Y. ZHENG, J. CHANG, Y. NIU, Y. SONG, K. GAI, Z. LI, D. JIN, Y. LI, ET AL., *Mixed attention network for cross-domain sequential recommendation*, in Proceedings of the 17th ACM International Conference on Web Search and Data Mining, 2024, pp. 405–413.

[142] J. LIN, R. MEN, A. YANG, C. ZHOU, Y. ZHANG, P. WANG, J. ZHOU, J. TANG, AND H. YANG, *M6: Multi-modality-to-multi-modality multitask mega-transformer for unified pretraining*, in Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2021, pp. 3251–3261.

[143] J. LIN, R. SHAN, C. ZHU, K. DU, B. CHEN, S. QUAN, R. TANG, Y. YU, AND W. ZHANG, *Rella: Retrieval-enhanced large language models for lifelong sequential behavior comprehension in recommendation*, arXiv preprint arXiv:2308.11131, (2023).

[144] Y. LIN, Y. LIU, F. LIN, L. ZOU, P. WU, W. ZENG, H. CHEN, AND C. MIAO, *A survey on reinforcement learning for recommender systems*, arXiv preprint arXiv:2109.10665, (2021).

[145] D. LIU, P. CHENG, H. ZHU, Z. DONG, X. HE, W. PAN, AND Z. MING, *Mitigating confounding bias in recommendation via information bottleneck*, in Proceedings of the 15th ACM conference on Recommender systems, 2021, pp. 351–360.

[146] J. LIU, C. LIU, R. LV, K. ZHOU, AND Y. ZHANG, *Is chatgpt a good recommender? a preliminary study*, ArXiv preprint, (2023).

[147] Q. LIU, Z. LIU, H. ZHANG, Y. CHEN, AND J. ZHU, *Mining cross features for financial credit risk assessment*, in Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 1069–1078.

[148] Q. LIU, X. WU, X. ZHAO, Y. WANG, Z. ZHANG, F. TIAN, AND Y. ZHENG, *Large language models enhanced sequential recommendation for long-tail user and item*, arXiv preprint arXiv:2405.20646, (2024).

[149] Q. LIU, Y. ZENG, R. MOKHOSI, AND H. ZHANG, *Stamp: short-term attention/memory priority model for session-based recommendation*, in Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 1831–1839.

[150] S. LIU AND Y. ZHENG, *Long-tail session-based recommendation*, in Proceedings of the 14th ACM conference on recommender systems, 2020, pp. 509–514.

[151] Y. LIU, Z. REN, W.-N. ZHANG, W. CHE, T. LIU, AND D. YIN, *Keywords generation improves e-commerce session-based recommendation*, in Proceedings of The Web Conference 2020, 2020, pp. 1604–1614.

[152] Y. LIU, X. ZHANG, M. ZOU, AND Z. FENG, *Co-occurrence embedding enhancement for long-tail problem in multi-interest recommendation*, in Proceedings of the 17th ACM Conference on Recommender Systems, 2023, pp. 820–825.

[153] M. LUDEWIG AND D. JANNACH, *Evaluation of session-based recommendation algorithms*, User Modeling and User-Adapted Interaction, 28 (2018), pp. 331–390.

[154] A. LUO, P. ZHAO, Y. LIU, F. ZHUANG, D. WANG, J. XU, J. FANG, AND V. S. SHENG, *Collaborative self-attention network for session-based recommendation.*, in IJCAI, 2020, pp. 2591–2597.

[155] M. MALEKIPIRBAZARI AND V. AKSAKALLI, *Risk assessment in social lending via random forests*, Expert Systems with Applications, 42 (2015), pp. 4621–4631.

[156] C. MEISTER, T. PIMENTEL, L. MALAGUTTI, E. WILCOX, AND R. COTTERELL, *On the efficacy of sampling adapters*, in Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2023, pp. 1437–1455.

[157] W. MENG, D. YANG, AND Y. XIAO, *Incorporating user micro-behaviors and item knowledge into multi-task learning for session-based recommendation*, in Proc. of SIGIR, 2020.

[158] N. MODANI, P. A. MITTAL, A. A. NANAVATI, AND B. SRIVASTAVA, *Series of dynamic targeted recommendations*, in International Conference on Electronic Commerce and Web Technologies, 2002.

[159] N. MODANI, Y. SABHARWAL, AND S. KARTHIK, *A framework for session based recommendations*, in International Conference on Electronic Commerce and Web Technologies, 2005.

[160] J. NARWARIYA, P. GUPTA, G. GUPTA, L. VIG, AND G. SHROFF, *X4sr: Post-hoc explanations for session-based recommendations.*, in Proc. of SIGIR, 2023.

[161] J. NI, J. LI, AND J. MCAULEY, *Justifying recommendations using distantly-labeled reviews and fine-grained aspects*, in Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP), 2019, pp. 188–197.

[162] C.-S. ONG, J.-J. HUANG, AND G.-H. TZENG, *Building credit scoring models using genetic programming*, Expert systems with applications, 29 (2005), pp. 41–47.

[163] K. OONO AND T. SUZUKI, *Graph neural networks exponentially lose expressive power for node classification*, arXiv preprint arXiv:1905.10947, (2019).

[164] M. OSKARSDOTTIR AND C. BRAVO, *Multilayer network analysis for improved credit risk prediction*, Omega, 105 (2021), p. 102520.

[165] K. OUYANG, X. XU, M. CHEN, Z. XIE, H.-T. ZHENG, S. SONG, AND Y. ZHAO, *Mining interest trends and adaptively assigning sample weight for session-based recommendation*, in Proc. of SIGIR, 2023, pp. 2174–2178.

[166] K. OUYANG, X. XU, C. TANG, W. CHEN, AND H. ZHENG, *Social-aware sparse attention network for session-based social recommendation*, in Proc. of EMNLP Findings, 2022.

[167] L. OUYANG, J. WU, X. JIANG, D. ALMEIDA, C. WAINWRIGHT, P. MISHKIN, C. ZHANG, S. AGARWAL, K. SLAMA, A. RAY, ET AL., *Training language models to follow instructions with human feedback*, Advances in neural information processing systems, 35 (2022), pp. 27730–27744.

[168] Z. PAN, F. CAI, W. CHEN, H. CHEN, AND M. DE RIJKE, *Star graph neural networks for session-based recommendation*, in Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 1195–1204.

[169] Z. PAN, F. CAI, Y. LING, AND M. DE RIJKE, *An intent-guided collaborative machine for session-based recommendation*, in Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 1833–1836.

[170] Y. PANG, L. WU, Q. SHEN, Y. ZHANG, Z. WEI, F. XU, E. CHANG, B. LONG, AND J. PEI, *Heterogeneous global graph neural networks for personalized session-based recommendation*, in Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, 2022, pp. 775–783.

[171] A. PEINTNER, A. R. MOHAMMADI, AND E. ZANGERLE, *Spare: Shortest path global item relations for efficient session-based recommendation*, in Proc. of RecSys, 2023.

[172] B. PEROZZI, R. AL-RFOU, AND S. SKIENA, *Deepwalk: Online learning of social representations*, in Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, 2014, pp. 701–710.

[173] S. PIRAMUTHU, *Financial credit-risk evaluation with neural and neurofuzzy systems*, European Journal of Operational Research, 112 (1999), pp. 310–321.

[174] M. POTTER, H. LIU, Y. LALA, C. LOANZON, AND Y. SUN, *Gru4recbe: A hybrid session-based movie recommendation system (student abstract)*, in Proc. of AAAI, 2022.

[175] S. QIAO, W. ZHOU, J. WEN, H. ZHANG, AND M. GAO, *Bi-channel multiple sparse graph attention networks for session-based recommendation*, in Proc. of CIKM, 2023, pp. 2075–2084.

[176] R. QIU, J. LI, Z. HUANG, AND H. YIN, *Rethinking the item order in session-based recommendation with graph neural networks*, in Proceedings of the 28th ACM international conference on information and knowledge management, 2019, pp. 579–588.

[177] R. QIU, H. YIN, Z. HUANG, AND T. CHEN, *Gag: Global attributed graph neural network for streaming session-based recommendation*, in Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 669–678.

[178] Z. QIU, Y. LI, P. NI, AND G. LI, *Credit risk scoring analysis based on machine learning models*, in 6th International Conference on Information Science and Control Engineering, 2019.

[179] M. QUADRANA, A. KARATZOGLOU, B. HIDASI, AND P. CREMONESI, *Personalizing session-based recommendations with hierarchical recurrent neural networks*, in

proceedings of the Eleventh ACM Conference on Recommender Systems, 2017, pp. 130–137.

[180] J. T. QUAH AND M. SRIGANESH, *Real-time credit card fraud detection using computational intelligence*, Expert systems with applications, 35 (2008), pp. 1721–1732.

[181] J. R. QUINLAN, *Induction of decision trees*, Machine learning, 1 (1986), pp. 81–106.

[182] A. RADFORD, J. W. KIM, C. HALLACY, A. RAMESH, G. GOH, S. AGARWAL, G. SASTRY, A. ASKELL, P. MISHKIN, J. CLARK, ET AL., *Learning transferable visual models from natural language supervision*, in International conference on machine learning, PMLR, 2021, pp. 8748–8763.

[183] R. RAFAILOV, A. SHARMA, E. MITCHELL, C. D. MANNING, S. ERMON, AND C. FINN, *Direct preference optimization: Your language model is secretly a reward model*, Advances in Neural Information Processing Systems, 36 (2024).

[184] P. RAVISANKAR, V. RAVI, G. R. RAO, AND I. BOSE, *Detection of financial statement fraud and feature selection using data mining techniques*, Decision support systems, 50 (2011), pp. 491–500.

[185] P. REN, Z. CHEN, J. LI, Z. REN, J. MA, AND M. DE RIJKE, *Repeatnet: A repeat aware neural recommendation machine for session-based recommendation*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 4806–4813.

[186] S. REN, Z. WU, AND K. Q. ZHU, *Emo: Earth mover distance optimization for auto-regressive language modeling*, arXiv preprint arXiv:2310.04691, (2023).

[187] S. RENDLE, C. FREUDENTHALER, Z. GANTNER, AND L. SCHMIDT-THIEME, *Bpr: Bayesian personalized ranking from implicit feedback*, in Proc. of UAI, 2009, pp. 452–461.

[188] S. RENDLE, C. FREUDENTHALER, AND L. SCHMIDT-THIEME, *Factorizing personalized markov chains for next-basket recommendation*, in Proceedings of the 19th international conference on World wide web, 2010, pp. 811–820.

[189] P. RESNICK AND H. R. VARIAN, *Recommender systems*, Communications of the ACM, 40 (1997), pp. 56–58.

[190] Y. RUBNER, C. TOMASI, AND L. J. GUIBAS, *The earth mover‚Äôs distance as a metric for image retrieval*, International Journal of Computer Vision, 40 (2000), pp. 99–121.

[191] S. SABOUR, N. FROSST, AND G. E. HINTON, *Dynamic routing between capsules*, Proc. of NeurIPS, (2017).

[192] R. SALAKHUTDINOV, A. MNIH, AND G. HINTON, *Restricted boltzmann machines for collaborative filtering*, in Proceedings of the 24th international conference on Machine learning, 2007, pp. 791–798.

[193] B. SARWAR, G. KARYPIS, J. KONSTAN, AND J. RIEDL, *Item-based collaborative filtering recommendation algorithms*, in Proceedings of the 10th international conference on World Wide Web, 2001, pp. 285–295.

[194] R. E. SCHAPIRE, *The strength of weak learnability*, Machine Learning, 5 (1990), pp. 197–227.

[195] J. J. SEOL, Y. KO, AND S.-G. LEE, *Exploiting session information in bert-based session-aware sequential recommendation*, in Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2022, pp. 2639–2644.

[196] C. SERRANO-CINCA, B. GUTIÉRREZ-NIETO, AND L. LÓPEZ-PALACIOS, *Determinants of default in p2p lending*, PloS one, 10 (2015), p. e0139427.

[197] W. SHALABY, S. OH, A. AFSHARINEJAD, S. KUMAR, AND X. CUI, *M2trec: Metadata-aware multi-task transformer for large-scale and cold-start free session-based recommendations*, in Proc. of RecSys, 2022.

[198] G. SHANI, D. HECKERMAN, R. I. BRAFMAN, AND C. BOUTILIER, *An mdp-based recommender system.*, Journal of Machine Learning Research, 6 (2005).

[199] B. SHAO, D. WANG, T. LI, AND M. OGIHARA, *Music recommendation based on acoustic features and user access patterns*, IEEE Transactions on Audio, Speech, and Language Processing, 17 (2009), pp. 1602–1611.

[200] Q. SHEN, L. WU, Y. PANG, Y. ZHANG, Z. WEI, F. XU, AND B. LONG, *Multi-behavior graph contextual aware network for session-based recommendation*, arXiv preprint arXiv:2109.11903, (2021).

[201] Q. SHEN, S. ZHU, Y. PANG, Y. ZHANG, AND Z. WEI, *Temporal aware multi-interest graph neural network for session-based recommendation*, in Asian Conference on Machine Learning, PMLR, 2023.

[202] I. SHENBIN, A. ALEKSEEV, E. TUTUBALINA, V. MALYKH, AND S. I. NIKOLENKO, *Recvae: A new variational autoencoder for top-n recommendations with implicit feedback*, in Proceedings of the 13th international conference on web search and data mining, 2020, pp. 528–536.

[203] S.-Y. SHIH AND H.-Y. CHI, *Automatic, personalized, and flexible playlist generation using reinforcement learning*, arXiv preprint arXiv:1809.04214, (2018).

[204] S. Y. SOHN AND J. W. KIM, *Decision tree-based technology credit scoring for start-up firms: Korean case*, Expert Systems with Applications, 39 (2012), pp. 4007–4012.

[205] B. SONG, Y. CAO, W. ZHANG, AND C. XU, *Session-based recommendation with hierarchical memory networks*, in Proc. of CIKM, 2019.

[206] J. SONG, C. MENG, AND S. ERMON, *Denoising diffusion implicit models*, arXiv preprint arXiv:2010.02502, (2020).

[207] W. SONG, Z. XIAO, Y. WANG, L. CHARLIN, M. ZHANG, AND J. TANG, *Session-based social recommendation via dynamic graph attention networks*, in Proceedings of the Twelfth ACM international conference on web search and data mining, 2019, pp. 555–563.

[208] X. SONG, J. LI, Q. LEI, W. ZHAO, Y. CHEN, AND A. MIAN, *Bi-clkt: Bi-graph contrastive learning based knowledge tracing*, Knowledge-Based Systems, 241 (2022), p. 108274.

[209] X. SONG, J. LI, Y. TANG, T. ZHAO, Y. CHEN, AND Z. GUAN, *Jkt: A joint graph convolutional network based deep knowledge tracing*, Information Sciences, 580 (2021), pp. 510–523.

[210] R. M. STEIN, *The relationship between default prediction and lending profits: Integrating roc analysis and loan pricing*, Journal of Banking & Finance, 29 (2005), pp. 1213–1236.

[211] J. SU, C. CHEN, W. LIU, F. WU, X. ZHENG, AND H. LYU, *Enhancing hierarchy-aware graph networks with deep dual clustering for session-based recommendation*, in Proc. of WWW, 2023.

[212] X. SU AND T. M. KHOSHGOFTAAR, *A survey of collaborative filtering techniques*, Adv. in Artif. Intell., 2009 (2009).

[213] C. SUN, H. YAN, X. QIU, AND X. HUANG, *Gaussian word embedding with a wasserstein distance loss*, arXiv preprint arXiv:1808.07016, (2018).

[214] F. SUN, J. LIU, J. WU, C. PEI, X. LIN, W. OU, AND P. JIANG, *Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer*, in Proc. of CIKM, 2019.

[215] H. SURYANTO, C. GUAN, A. VOUMARD, AND G. BEYDOUN, *Transfer learning in credit risk.*, in Joint European Conference on Machine Learning and Knowledge Discovery in Databases, 2019, pp. 483–498.

[216] F. TAN, X. HOU, J. ZHANG, Z. WEI, AND Z. YAN, *A deep learning approach to competing risks representation in peer-to-peer lending*, IEEE transactions on neural networks and learning systems, 30 (2018), pp. 1565–1574.

[217] Y. K. TAN, X. XU, AND Y. LIU, *Improved recurrent neural networks for session-based recommendations*, in Proceedings of the 1st workshop on deep learning for recommender systems, 2016, pp. 17–22.

[218] J. TANG AND K. WANG, *Personalized top-n sequential recommendation via convolutional sequence embedding*, in Proceedings of the eleventh ACM international conference on web search and data mining, 2018, pp. 565–573.

[219] Y. TIAN, J. CHANG, Y. NIU, Y. SONG, AND C. LI, *When multi-level meets multi-interest: A multi-grained neural model for sequential recommendation*, in Proc. of SIGIR, 2022, pp. 1632–1641.

[220] H. TOUVRON, T. LAVRIL, G. IZACARD, X. MARTINET, M.-A. LACHAUX, T. LACROIX, B. ROZIÈRE, N. GOYAL, E. HAMBRO, F. AZHAR, ET AL., *Llama: Open and efficient foundation language models*, arXiv preprint arXiv:2302.13971, (2023).

[221] H. TURGUT, T. D. YETKI, Ö. BALI, AND T. A. YÜCEL, *Prod2vec-var: A session based recommendation system with enhanced diversity*, in Proc. of CIKM, 2023, pp. 5253–5254.

[222] M. S. UDDIN, G. CHI, M. A. AL JANABI, AND T. HABIB, *Leveraging random forest in micro-enterprises credit risk modelling for accuracy and interpretability*, International Journal of Finance and Economics, (2020).

[223] S. VARGAS AND P. CASTELLS, *Rank and relevance in novelty and diversity metrics for recommender systems*, in Proceedings of the fifth ACM conference on Recommender systems, 2011, pp. 109–116.

[224] A. VASWANI, N. SHAZEER, N. PARMAR, J. USZKOREIT, L. JONES, A. N. GOMEZ, Ł. KAISER, AND I. POLOSUKHIN, *Attention is all you need*, Advances in neural information processing systems, 30 (2017).

[225] P. VELIČKOVIĆ, G. CUCURULL, A. CASANOVA, A. ROMERO, P. LIO, AND Y. BENGIO, *Graph attention networks*, arXiv preprint arXiv:1710.10903, (2017).

[226] L. VILNIS AND A. MCCALLUM, *Word representations via gaussian embedding*, arXiv preprint arXiv:1412.6623, (2014).

[227] D. WANG, Z. ZHANG, J. ZHOU, P. CUI, J. FANG, Q. JIA, Y. FANG, AND Y. QI, *Temporal-aware graph neural network for credit risk prediction*, in Proceedings of the 2021 SIAM International Conference on Data Mining (SDM), SIAM, 2021, pp. 702–710.

[228] J. WANG, A. P. DE VRIES, AND M. J. REINDERS, *Unifying user-based and item-based collaborative filtering approaches by similarity fusion*, in Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, 2006, pp. 501–508.

[229] J. WANG, K. DING, Z. ZHU, AND J. CAVERLEE, *Session-based recommendation with hypergraph attention networks*, in Proceedings of the 2021 SIAM international conference on data mining (SDM), SIAM, 2021, pp. 82–90.

[230] L. WANG AND E.-P. LIM, *Zero-shot next-item recommendation using large pretrained language models*, arXiv preprint arXiv:2304.03153, (2023).

[231] L. WANG, X. XU, K. OUYANG, H. DUAN, Y. LU, AND H.-T. ZHENG, *Self-supervised dual-channel attentive network for session-based social recommendation*, in 2022 IEEE 38th International Conference on Data Engineering (ICDE), 2022.

[232] M. WANG, P. REN, L. MEI, Z. CHEN, J. MA, AND M. DE RIJKE, *A collaborative session-based recommendation approach with parallel memory modules*, in Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, 2019, pp. 345–354.

[233] P. WANG, Y. FAN, L. XIA, W. X. ZHAO, S. NIU, AND J. HUANG, *Kerl: A knowledge-guided reinforcement learning model for sequential recommendation*, in Proc. of SIGIR, 2020.

[234] S. WANG, L. CAO, Y. WANG, Q. Z. SHENG, M. A. ORGUN, AND D. LIAN, *A survey on session-based recommender systems*, ACM Computing Surveys (CSUR), 54 (2021), pp. 1–38.

[235] S. WANG, L. HU, AND L. CAO, *Perceiving the next choice with comprehensive transaction embeddings for online recommendation*, in Proc. of ECML, 2017.

[236] T. WANG AND P. ISOLA, *Understanding contrastive representation learning through alignment and uniformity on the hypersphere*, in International Conference on Machine Learning, PMLR, 2020, pp. 9929–9939.

[237] W. WANG, F. FENG, X. HE, X. WANG, AND T.-S. CHUA, *Deconfounded recommendation for alleviating bias amplification*, in Proc. of KDD, 2021.

[238] W. WANG, Y. XU, F. FENG, X. LIN, X. HE, AND T.-S. CHUA, *Diffusion recommender model*, in Proc. of SIGIR, 2023, pp. 832–841.

[239] W. WANG, H. YIN, Z. HUANG, Q. WANG, X. DU, AND Q. V. H. NGUYEN, *Streaming ranking based recommender systems*, in Proc. of SIGIR, 2018.

[240] W. WANG, W. ZHANG, S. LIU, Q. LIU, B. ZHANG, L. LIN, AND H. ZHA, *Beyond clicks: Modeling multi-relational item graph for session-based target behavior prediction*, in Proceedings of The Web Conference 2020, 2020, pp. 3056–3062.

[241] X. WANG, H. JI, C. SHI, B. WANG, Y. YE, P. CUI, AND P. S. YU, *Heterogeneous graph attention network*, in Proc. of WWW, 2019.

[242] Y. WANG, Z. LIU, Y. WANG, X. ZHAO, B. CHEN, H. GUO, AND R. TANG, *Diff-msr: A diffusion model enhanced paradigm for cold-start multi-scenario recommendation*, in Proc. of WSDM, 2024, pp. 779–787.

[243] Y. WANG AND X. S. NI, *Risk prediction of peer-to-peer lending market by a lstm model with macroeconomic factor*, in ACM Southeast Conference, 2020, pp. 181–187.

[244] Y. WANG, S. WANG, AND K. K. LAI, *A new fuzzy support vector machine to evaluate credit risk*, IEEE Transactions on Fuzzy Systems, 13 (2005), pp. 820–831.

[245] Z. WANG, C. CHEN, K. ZHANG, Y. LEI, AND W. LI, *Variational recurrent model for session-based recommendation*, in Proc. of CIKM, 2018.

[246] Z. WANG, W. WEI, G. CONG, X.-L. LI, X.-L. MAO, AND M. QIU, *Global context enhanced graph neural networks for session-based recommendation*, in Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 169–178.

[247] Z. WANG, G. WU, AND Y. WANG, *Effectively using long and short sessions for multi-session-based recommendations*, arXiv preprint arXiv:2205.04366, (2022).

[248] C. WEI, B. BAI, K. BAI, AND F. WANG, *Gsl4rec: Session-based recommendations with collective graph structure learning and next interaction prediction*, in Proc. of WWW, 2022.

[249] S. WEI, J. LV, Y. GUO, Q. YANG, X. CHEN, Y. ZHAO, Q. LI, F. ZHUANG, AND G. KOU, *Combining intra-risk and contagion risk for enterprise bankruptcy prediction using graph neural networks*, Information Sciences, (2024), p. 120081.

[250] T. WEI, F. FENG, J. CHEN, Z. WU, J. YI, AND X. HE, *Model-agnostic counterfactual reasoning for eliminating popularity bias in recommender system*, in Proc. of KDD, 2021.

[251] T. WILM, P. NORMANN, S. BAUMEISTER, AND P.-V. KOBOW, *Scaling session-based transformer recommendations using optimized negative sampling and loss functions*, in Proc. of RecSys, 2023.

[252] C. WU, F. WU, T. QI, AND Y. HUANG, *Mm-rec: multimodal news recommendation*, arXiv preprint arXiv:2104.07407, (2021).

[253] C. WU AND M. YAN, *Session-aware information embedding for e-commerce product recommendation*, in Proceedings of the 2017 ACM on conference on information and knowledge management, 2017.

[254] F. WU, A. SOUZA, T. ZHANG, C. FIFTY, T. YU, AND K. WEINBERGER, *Simplifying graph convolutional networks*, in International conference on machine learning, PMLR, 2019, pp. 6861–6871.

[255] H. WU, C. GENG, AND H. FANG, *Causality and correlation graph modeling for effective and explainable session-based recommendation*, ACM Transactions on the Web, 18 (2023), pp. 1–25.

[256] S. WU, F. SUN, W. ZHANG, X. XIE, AND B. CUI, *Graph neural networks in recommender systems: a survey*, ACM Computing Surveys, (2022), pp. 1–37.

[257] S. WU, Y. TANG, Y. ZHU, L. WANG, X. XIE, AND T. TAN, *Session-based recommendation with graph neural networks*, in Proceedings of the AAAI conference on artificial intelligence, vol. 33, 2019, pp. 346–353.

[258] T.-C. WU AND M.-F. HSU, *Credit risk assessment and decision making by a fusion approach*, Knowledge-Based Systems, 35 (2012), pp. 102–110.

[259] Z. WU, S. PAN, G. LONG, J. JIANG, X. CHANG, AND C. ZHANG, *Connecting the dots: Multivariate time series forecasting with graph neural networks*, in Proc. of KDD, 2020.

[260] X. XIA, H. YIN, J. YU, Y. SHAO, AND L. CUI, *Self-supervised graph co-training for session-based recommendation*, in Proceedings of the 30th ACM international conference on information & knowledge management, 2021, pp. 2180–2190.

[261] X. XIA, H. YIN, J. YU, Q. WANG, L. CUI, AND X. ZHANG, *Self-supervised hypergraph convolutional networks for session-based recommendation*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 4503–4511.

[262] X. XIA, J. YU, Q. WANG, C. YANG, N. Q. V. HUNG, AND H. YIN, *Efficient on-device session-based recommendation*, ACM Transactions on Information Systems, 41 (2023), pp. 1–24.

[263] X. XIA, J. YU, G. XU, AND H. YIN, *Towards communication-efficient model updating for on-device session-based recommendation*, in Proc. of CIKM, 2023, pp. 2795–2804.

[264] Y. XIAN, Z. FU, S. MUTHUKRISHNAN, G. DE MELO, AND Y. ZHANG, *Reinforcement knowledge graph reasoning for explainable recommendation*, in Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, 2019, pp. 285–294.

[265] X. XIE, F. SUN, Z. LIU, S. WU, J. GAO, J. ZHANG, B. DING, AND B. CUI, *Contrastive learning for sequential recommendation*, in 2022 IEEE 38th international conference on data engineering (ICDE), IEEE, 2022, pp. 1259–1273.

[266] Y. XIE, P. ZHOU, AND S. KIM, *Decoupled side information fusion for sequential recommendation*, in Proc. of SIGIR, 2022, pp. 1611–1621.

[267] X. XIN, L. YANG, Z. ZHAO, P. REN, Z. CHEN, J. MA, AND Z. REN, *On the effectiveness of unlearning in session-based recommendation*, in Proceedings of the 17th ACM International Conference on Web Search and Data Mining, 2024, pp. 855–863.

[268] C. XU, P. ZHAO, Y. LIU, V. S. SHENG, J. XU, F. ZHUANG, J. FANG, AND X. ZHOU, *Graph contextualized self-attention network for session-based recommendation.*, in IJCAI, vol. 19, 2019, pp. 3940–3946.

[269] S. XU, J. TAN, S. HEINECKE, V. J. LI, AND Y. ZHANG, *Deconfounded causal collaborative filtering*, ACM Transactions on Recommender Systems, (2023), pp. 1–25.

[270] A. YANG, N. WANG, H. DENG, AND H. WANG, *Explanation as a defense of recommendation*, in Proceedings of the 14th ACM International Conference on Web Search and Data Mining, 2021, pp. 1029–1037.

[271] H. YANG, Y. CHOI, G. KIM, AND J.-H. LEE, *Loam: Improving long-tail session-based recommendation via niche walk augmentation and tail session mixup*, in Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2023, pp. 527–536.

[272] L. YANG, Z. ZHANG, Y. SONG, S. HONG, R. XU, Y. ZHAO, W. ZHANG, B. CUI, AND M.-H. YANG, *Diffusion models: A comprehensive survey of methods and applications*, ACM Computing Surveys, 56 (2023), pp. 1–39.

[273] M. YANG, Q. DAI, Z. DONG, X. CHEN, X. HE, AND J. WANG, *Top-n recommendation with counterfactual user preference simulation*, in Proc. of CIKM, 2021.

[274] Y. YANG, *Adaptive credit scoring with kernel learning methods*, European Journal of Operational Research, 183 (2007), pp. 1521–1536.

[275] Y. YANG, J. ZHANG, Y. WANG, Z. MIAO, AND Y. TONG, *Multiple connectivity views for session-based recommendation*, in Proceedings of the 17th ACM Conference on Recommender Systems, 2023, pp. 1000–1006.

[276] G.-E. YAP, X.-L. LI, AND P. S. YU, *Effective next-items recommendation via personalized sequential pattern mining*, in Proc. of DASFAA, 2012.

[277] R. YE, Q. ZHANG, AND H. LUO, *Cross-session aware temporal convolutional network for session-based recommendation*, in 2020 International Conference on Data Mining Workshops (ICDMW), IEEE, 2020, pp. 220–226.

[278] I.-C. YEH AND C.-H. LIEN, *The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients*, Expert Systems with Applications, 36 (2009), pp. 2473–2480.

[279] Q. YIN, H. FANG, Z. SUN, AND Y.-S. ONG, *Understanding diversity in session-based recommendation*, ACM Transactions on Information Systems, 42 (2023), pp. 1–34.

[280] H. YING, J. WU, G. XU, Y. LIU, T. LIANG, X. ZHANG, AND H. XIONG, *Time-aware metric embedding with asymmetric projection for successive poi recommendation*, World Wide Web, 22 (2019), pp. 2209–2224.

[281] H. YING, F. ZHUANG, F. ZHANG, Y. LIU, G. XU, X. XIE, H. XIONG, AND J. WU, *Sequential recommender system based on hierarchical attention network*, in IJCAI International Joint Conference on Artificial Intelligence, 2018.

[282] R. YING, R. HE, K. CHEN, P. EKSOMBATCHAI, W. L. HAMILTON, AND J. LESKOVEC, *Graph convolutional neural networks for web-scale recommender systems*, in Proc. of KDD, 2018.

[283] D. YU, Q. LI, H. YIN, AND G. XU, *Causality-guided graph learning for session-based recommendation*, in Proc. of CIKM, 2023, pp. 3083–3093.

[284] F. YU, Y. ZHU, Q. LIU, S. WU, L. WANG, AND T. TAN, *Tagnn: Target attentive graph neural networks for session-based recommendation*, in Proc. of SIGIR, 2020.

[285] F. YUAN, X. HE, H. JIANG, G. GUO, J. XIONG, Z. XU, AND Y. XIONG, *Future data helps training: Modeling future contexts for session-based recommendation*, in Proceedings of The Web Conference 2020, 2020, pp. 303–313.

[286] J. YUAN, W. JI, D. ZHANG, J. PAN, AND X. WANG, *Micro-behavior encoding for session-based recommendation*, in 2022 IEEE 38th International Conference on Data Engineering (ICDE), 2022.

[287] J. YUAN, Z. SONG, M. SUN, X. WANG, AND W. X. ZHAO, *Dual sparse attention network for session-based recommendation*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 4635–4643.

[288] E. ZANGERLE, M. PICHL, W. GASSLER, AND G. SPECHT, # *nowplaying music dataset: Extracting listening behavior from twitter*, in Proceedings of the first international workshop on internet-scale multimedia management, 2014.

[289] H. ZENG, Z. YUE, Q. JIANG, AND D. WANG, *Federated recommendation via hybrid retrieval augmented generation*, arXiv preprint arXiv:2403.04256, (2024).

[290] M. ZHANG, S. WU, M. GAO, X. JIANG, K. XU, AND L. WANG, *Personalized graph neural networks with attention mechanism for session-aware recommendation*, IEEE Transactions on Knowledge and Data Engineering, 34 (2020), pp. 3946–3957.

[291] P. ZHANG, J. GUO, C. LI, Y. XIE, J. B. KIM, Y. ZHANG, X. XIE, H. WANG, AND S. KIM, *Efficiently leveraging multi-level user intent for session-based recommendation via atten-mixer network*, in Proc. of WSDM, 2023.

[292] R. ZHANG, Y. GU, X. SHEN, AND H. SU, *Knowledge-enhanced session-based recommendation with temporal transformer*, arXiv preprint arXiv:2112.08745, (2021).

[293] X. ZHANG, B. XU, F. MA, C. LI, L. YANG, AND H. LIN, *Beyond co-occurrence: Multi-modal session-based recommendation*, IEEE Transactions on Knowledge and Data Engineering, (2023).

[294] X. ZHANG, B. XU, L. YANG, C. LI, F. MA, H. LIU, AND H. LIN, *Price does matter! modeling price and interest preferences in session-based recommendation*, in Proc. of SIGIR, 2022, pp. 1684–1693.

[295] Y. ZHANG, X. CHEN, ET AL., *Explainable recommendation: A survey and new perspectives*, Foundations and Trends® in Information Retrieval, 14 (2020), pp. 1–101.

[296] Y. ZHANG, F. FENG, X. HE, T. WEI, C. SONG, G. LING, AND Y. ZHANG, *Causal intervention for leveraging popularity bias in recommendation*, in Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval, 2021, pp. 11–20.

[297] Y. ZHANG, G. LAI, M. ZHANG, Y. ZHANG, Y. LIU, AND S. MA, *Explicit factor models for explainable recommendation based on phrase-level sentiment analysis*, in Proc. of SIGIR, 2014, pp. 83–92.

[298] Z. ZHANG AND O. NASRAOUI, *Efficient hybrid web recommendations based on markov clickstream models and implicit search*, in IEEE/WIC/ACM International Conference on Web Intelligence (WI'07), 2007.

[299] Z. ZHANG, A. WANG, Y. ZHANG, Y. REN, W. LI, B. WANG, AND M. INUIGUCHI, *Relation pruning and discriminative sampling over knowledge graph for long-tail recommendation*, Information Sciences, (2024), p. 120871.

[300] Z. ZHANG AND B. WANG, *Graph neighborhood routing and random walk for session-based recommendation*, in 2021 IEEE International Conference on Data Mining (ICDM), IEEE, 2021, pp. 1517–1522.

[301] ——, *Graph spring network and informative anchor selection for session-based recommendation*, Neural Networks, (2023), pp. 43–56.

[302] H. ZHAO, Y. GE, Q. LIU, G. WANG, E. CHEN, AND H. ZHANG, *P2p lending survey: platforms, recent advances and prospects*, ACM Transactions on Intelligent Systems and Technology (TIST), 8 (2017), pp. 1–28.

[303] H. ZHAO, Q. LIU, G. WANG, Y. GE, AND E. CHEN, *Portfolio selections in p2p lending: A multi-objective perspective*, in Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, 2016, pp. 2075–2084.

[304] Y. ZHENG, C. GAO, X. LI, X. HE, Y. LI, AND D. JIN, *Disentangling user interest and conformity for recommendation with causal embedding*, in Proc. of WWW, 2021.

[305] Y. ZHENG, S. LIU, Z. LI, AND S. WU, *Dgtn: Dual-channel graph transition network for session-based recommendation*, in 2020 International Conference on Data Mining Workshops (ICDMW), IEEE, 2020, pp. 236–242.

[306] Q. ZHONG, Y. LIU, X. AO, B. HU, J. FENG, J. TANG, AND Q. HE, *Financial defaulter detection on online credit payment via multi-view attributed heterogeneous information network*, in Proceedings of The Web Conference 2020, 2020, pp. 785–795.

[307] F. ZHOU, X. QI, C. XIAO, AND J. WANG, *Metarisk: Semi-supervised few-shot operational risk classification in banking industry*, Information Sciences, 552 (2021), pp. 1–16.

[308] F. ZHOU, Z. WEN, K. ZHANG, G. TRAJCEVSKI, AND T. ZHONG, *Variational session-based recommendation using normalizing flows*, in Proc. of WWW, 2019.

[309] H. ZHOU, Q. TAN, X. HUANG, K. ZHOU, AND X. WANG, *Temporal augmented graph neural networks for session-based recommendations*, in Proc. of SIGIR, 2021.

[310] K. ZHOU, H. WANG, W. X. ZHAO, Y. ZHU, S. WANG, F. ZHANG, Z. WANG, AND J.-R. WEN, *S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization*, in Proceedings of the 29th ACM international conference on information & knowledge management, 2020, pp. 1893–1902.

[311] X. ZHOU, W. ZHANG, AND Y. JIANG, *Personal credit default prediction model based on convolution neural network*, Mathematical Problems in Engineering, (2020).

[312] G. ZHU, H. HOU, J. CHEN, C. YUAN, AND Y. HUANG, *Transition relation aware self-attention for session-based recommendation*, arXiv preprint arXiv:2203.06407, (2022).

[313] L. ZHU, D. QIU, D. ERGU, C. YING, AND K. LIU, *A study on predicting loan default based on the random forest algorithm*, Procedia Computer Science, 162 (2019), pp. 503–513.

[314] Z. ZHU, C. WU, R. FAN, D. LIAN, AND E. CHEN, *Membership inference attacks against sequential recommender systems*, in Proc. of WWW, 2023, pp. 1208–1219.