成都理工大學 中英合作办学
CDUT Sino-British Collaborative Education

OXFORD BROOKES UNIVERSITY

UNDERGRADUATE PROJECT PROGRESS REPORT

Project Title:	**Federated Convolutional Generative Network for Next Item Recommendation**
Surname:	**Long**
First Name:	**Yicheng(Jack)**
Student Number:	**202018010116**
Supervisor Name:	**Joojo Walker**
Module Code:	**CHC 6096**
Module Name:	**Project**
Date Submitted:	**2023.12.25**

Table of Contents

1 Introduction

1.1 Background

In recent years, in the field of recommendation systems, the Next Item Recommendation System has gained increasing attention due to its ability to infer dynamic user preferences through sequential user interactions, as well as its real-time and personalized advantages. Especially in applications that focus on real-time or continuous user experiences, the Next Item Recommendation System has become very popular. For example, users of Last.fm or Weishi typically enjoy a series of songs/videos over a period of time [1]. Convolutional neural network (CNN), as a deep learning neural network architecture, has excellent feature extraction ability and sequence data processing performance. Therefore, the application of CNN in recommendation systems is very common. In addition, federated learning, as a decentralized machine learning method, allows models to be trained on local devices in recommendation systems, thereby protecting user data privacy and reducing data transportation costs. It is now widely used in recommendation systems.

1.2 Aim

This project is the next recommendation system based on federated learning and convolutional generative networks. This system solves the user data privacy and data transmission cost issues of traditional recommendation systems by introducing federated learning, and processes data by introducing convolutional generation networks. Finally, a recommendation system that combines the above advantages can surpass existing recommendation systems to achieve better recommendation results.

1.3 Objectives

• Collection of relevant literature

• Evaluation and Comparison of Models

Evaluate and compare the advantages and disadvantages of different federated learning recommendation systems and the next recommendation model based on CNN.

• Improved model

Propose an improvement to the existing next recommendation model based on CNN, introducing federated learning to the recommendation model.

• Selection and Preprocessing of Datasets
Select a suitable dataset from public resources and preprocess the data to meet the input requirements of the model.

• Implementation of the model
Implement a federated learning framework through programming, based on the next recommendation model of CNN, and then combine them.

• Experimental testing and optimization
Use the dataset as input to the recommendation system, conduct repeated testing, and adjust parameters and models according to different situations.

• Analysis and Summary
Analyze and summarize the experimental results, evaluate the performance of the recommendation system, including advantages and disadvantages, and complete the paperwork.

• Presentation Prepare
Create PowerPoint presentations, demonstrate videos, and prepare speeches.

1.4 Project Overview

1.4.1 Scope

The purpose of the study is to improve the next recommendation system by introducing a federated convolutional generation network model to overcome the shortcomings of traditional CNN recommendation systems. This model aims to better capture the features and sequential patterns in user project interaction sequences, thereby providing more personalized and accurate recommendations. Regarding importance, it has technological innovation and addresses the shortcomings of traditional methods. It also has significant importance for practical applications, improving user experience and having a positive impact in commercial applications.

1.4.2 Audience

Users and producers of products that require real-time recommendations will be one of the main beneficiaries, such as applications in music streaming, video on demand, e-commerce, online games, and other fields. For their product users, the Next Item Recommendation System will provide more personalized and real-time recommendations that users are interested in, making it easier for product users to find products, services, or information of interest, This saves time and effort, and product developers gain more users as a result. In addition, some companies or advertisers are the main beneficiaries. By increasing transaction volume, businesses can benefit, and advertisers can place more targeted advertisements to improve advertising efficiency. Technical researchers will also benefit from new technologies and methods.

2 Background Review

The earliest work and ideas for sequence recommendation mainly relied on Markov chains [2] and feature based matrix decomposition [3] methods. Markov chains are a mathematical model in which the occurrence of an event only depends on the state of the previous event, and is independent of the earlier state. But it also has some shortcomings, especially when dealing with complex sequence data, its ability to model complex nonlinear relationships and patterns in sequence data is limited and lacks long-term memory. Afterwards, deep learning models gradually began to demonstrate advanced recommendation accuracy. In 2016, Hidasi et al. [4] proposed a DL based SBR system, commonly known as GRU4Rec. This is the first model to use RNN, which introduces session parallel small batch, output sampling based on small batch, and sorting loss function, resulting in significant results due to popular baselines. In 2018, Tang and Wang [5] proposed a new sequence recommendation called Caser. They abandoned the RNN structure and proposed a convolutional sequence embedding model, demonstrating that this CNN based recommendation can achieve similar or superior performance in the popular RNN model's top-N sequence recommendation. Not long after the same year, Yuan et al. [1] proposed a simple, efficient, and efficient convolutional generation model for session based top-N project recommendations. This model is suitable for short-term and long-term project dependencies and simplifies deeper network optimization. Ultimately, the model's recommendation accuracy and effectiveness are significantly better than existing technologies at the time. In 2021, Song et al. [6] designed an effective SBRS called Intersessional Collaborative

Recommendation Network (Insert) to recommend the next project in short sessions, and designed a Session Retrieval Network (SSRN) to identify sessions similar to the current short session from the historical sessions of the current user and other users, resulting in better recommendation performance than the most advanced series of recommendations at the time. In the same year, Chai et al. [7] proposed a secure matrix decomposition framework under federated learning settings, called FedMF, which to some extent prevents users' raw data leakage and ensures user data privacy, but has not been applied in recommendation systems. In 2023, Li et al. [8] proposed Federated Recommendation with Additive Personalization (FedRAP), which introduced federated learning. This recommendation system effectively avoids user information leakage, reduces communication costs, and solves the problem of poor personalization in other federated recommendations. Afterwards, Kumar et al. [9] proposed a Horizontal Vertical Convolutional Neural Network (HV-CNN) embedded with Word2Vec technology, which outperformed state-of-the-art methods on 30 publicly available music datasets.

3 Methodology

3.1 Approach

3.1.1 Federated Learning Recommendation System

Federated learning is a privacy preserving distributed learning scheme proposed by Google, which allows machine learning models to use intermediate model parameters for training, avoiding direct use of user real data and thus protecting user privacy. Due to the fact that traditional recommendation systems typically store user data centrally on a server and use it directly for training and testing, introducing federated learning into recommendation systems can effectively protect user privacy data. In this article, we refer to it as the Federated Recommendation System (FedRS). FedRS uses a federated learning architecture based on local storage of participant data, as shown in Figure 1. Its communication architecture can be divided into client server architecture and peer architecture. The client server architecture relies on a central server to perform initialization and model aggregation tasks. The server is responsible for distributing the global recommendation model to selected clients, and then the clients use the received model and local data for local training in each round. Finally, the client sends the updated intermediate parameters (such as model parameters and gradients) back to the server for global model aggregation. The peer-to-peer architecture does not have a central server participating in the communication process. In each round of communication, each participant broadcasts the updated intermediate parameters to some random online neighbors, and then aggregates the received parameters into their own global model. This architecture can avoid single point of failure issues and privacy issues related to central servers.

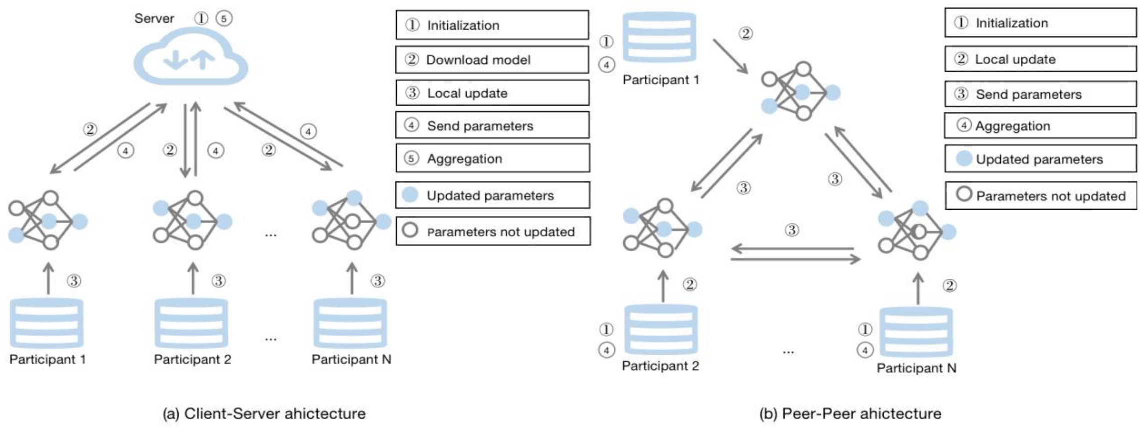


Figure 1: Communication architecture of FedRS

3.1.2 A Convolutional Generation Model for Item Recommendation Based on Conversation

This model directly processes the sequence of items that have been previously interacted with. Its goal is to estimate the distribution of the original item interaction sequence in order to reliably calculate the probability of the item and generate future items that users may enjoy interacting with. The model introduces a probability distribution p (x), which represents the joint distribution of item sequence x={x0,..., xt}. To model p (x), the chain rule can be used to decompose it into a product of a series of conditional distributions, taking into account the probability distribution of each item, as shown in Figure 2:

$$p(\mathbf{x}) = \prod_{i=1}^{t} p(x_i | x_{0:i-1}, \boldsymbol{\theta}) p(x_0)$$

Figure 2: Conditional probability of an item

Next, K proposes to use an extended 1D convolutional neural network (CNN) to model the conditional distribution of user item interaction, as shown in Figure 3 (b). Compared with the standard CNN (Figure 3 (a)), the extended convolutional neural network uses the convolutional operation of "l-dialated convolution" to obtain a larger receptive field without introducing more parameters. This enables the network to better capture long-range dependencies in input data.

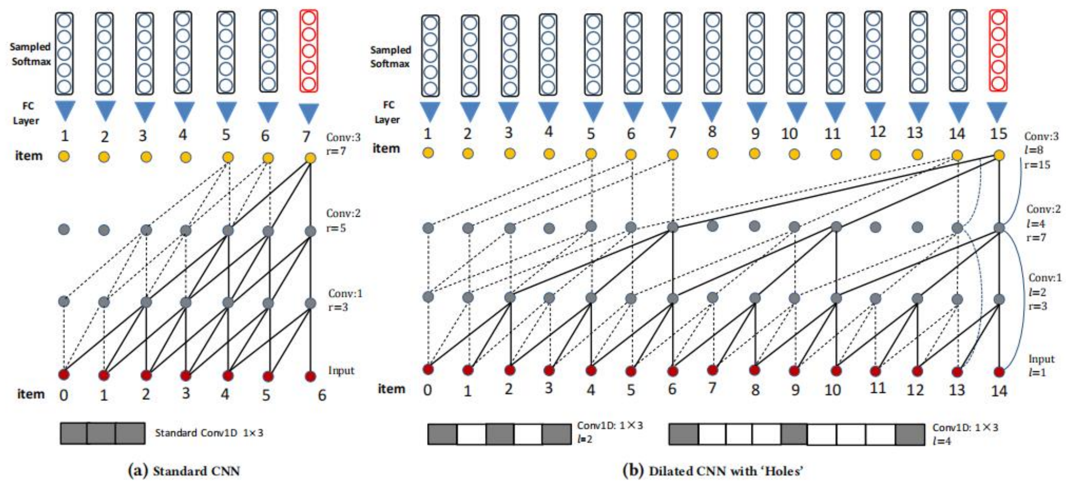


Figure 3: Conditional distribution prediction

3.2 Technology

Hardware includes: computer (GPU: NVIDIA RTX2070S, RAM: 500G), software includes: deep learning framework: TensorFlow, programming language: Python 3.9, editor: PyCharm3.3, data preprocessing tool: MySQL.

3.3 Testing and Evaluation Plan

3.3.1 Data Testing

Check if the data type is correct and ensure that the data format meets expectations. (For example, for the Yoochoosebuys dataset, there should be four columns, namely: session id: token, item_id: token, count: float, timestamp: float)

Ensure the quality and consistency of the dataset. Check if the data is complete and if there are any missing or abnormal values.

Data preprocessing, the dataset used this time comes from the preprocessed dataset provided by Rebole.

3.3.2 Model Testing

Pre train Testing:

Environmental configuration testing: Ensure that the RecBole framework is installed and configured correctly on your system.

Input data format test: Ensure that the dataset can be loaded correctly into the NextItNet model.

Post train Testing:

Data transformation test: Transform input data (such as data augmentation, noise injection, etc.) to check the robustness of the model to data transformation.

Performance evaluation:

Recommendation performance indicators: Use appropriate indicators (such as accuracy, recall, F1 value, etc.) to evaluate and compare the recommendation performance of the model on different datasets.

Resource utilization evaluation: measures the computational resource consumption of the model, such as runtime, memory usage, etc.

3.4 Design and Implementation

In terms of research topic, the author has currently completed the understanding of the model of the template model, and after completing the literature review, has gained a new understanding of the field of recommendation models.

In terms of the dataset, the author completed the download and testing of the dataset, as well as the matching between the dataset and the model.

In terms of model implementation, the author has completed the configuration of the environment and learned how to use the Rebole tool to test and train the model. The model used this time, NextItNet, was implemented and tested using Rebole, and the training results on the built-in dataset and YOOCHOOSE were compared.

Model test results

Figure 4 shows the result of quic start using NextItNet on the recoble, which is the training result using the built-in dataset: ml-100k.

Explanation of indicators:

recall@10 (Recall rate): measures how many items users actually like in the top 10 positions of the recommendation list in the model. The calculation method is the ratio of the actual number of recommended favorite items to the total number of favorite items.

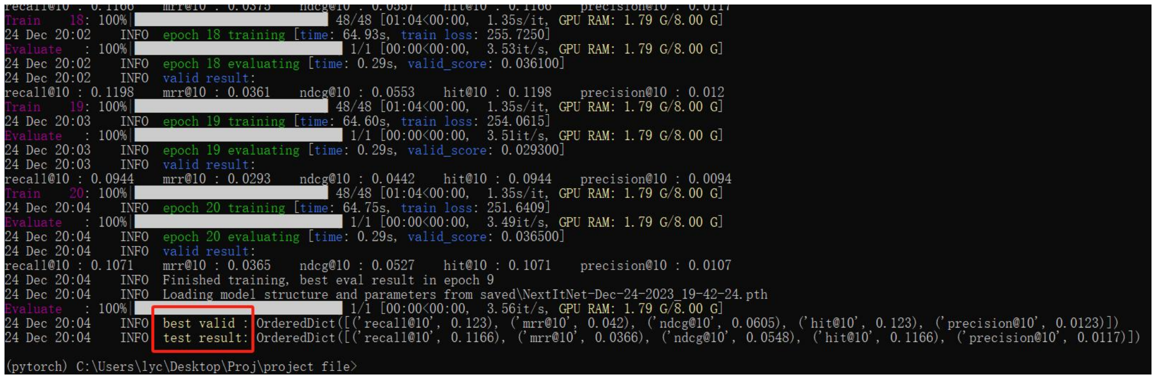
mrr@10 (Average reciprocal ranking): measures the average reciprocal of the items actually liked by the highest ranked user in the recommendation list. Countdown is to emphasize items with higher rankings. The calculation method is to take the average of the reciprocal rankings of the items that each user actually likes.

ndcg@10 (Normalized cumulative loss gain): measures the ranking quality of items in the recommended list of the model. For each user, calculate the cumulative gain of each item in the recommendation list, normalize it, and finally take the average to obtain the overall indicator.

hit@10 (Hit rate): measures how many hits the model has in the top 10 recommendations (the items that the user actually likes are in the recommendation list). The calculation method is the ratio of the actual number of recommended favorite items to the total number of hit items.

precision@10 (Accuracy): Measures how many items the model actually likes among the top 10 recommendations. The calculation method is the ratio of the actual number of items recommended and liked by the user to the total number of recommended items.

The results show an average training time of 64 seconds per round, with a training loss of approximately 250. The results on the test set are recall@10 =0.1166, mrr@10 =0.0366, ndcg@10 =0.0548, hit@10 =0.1166, precision@10 =0.0117.

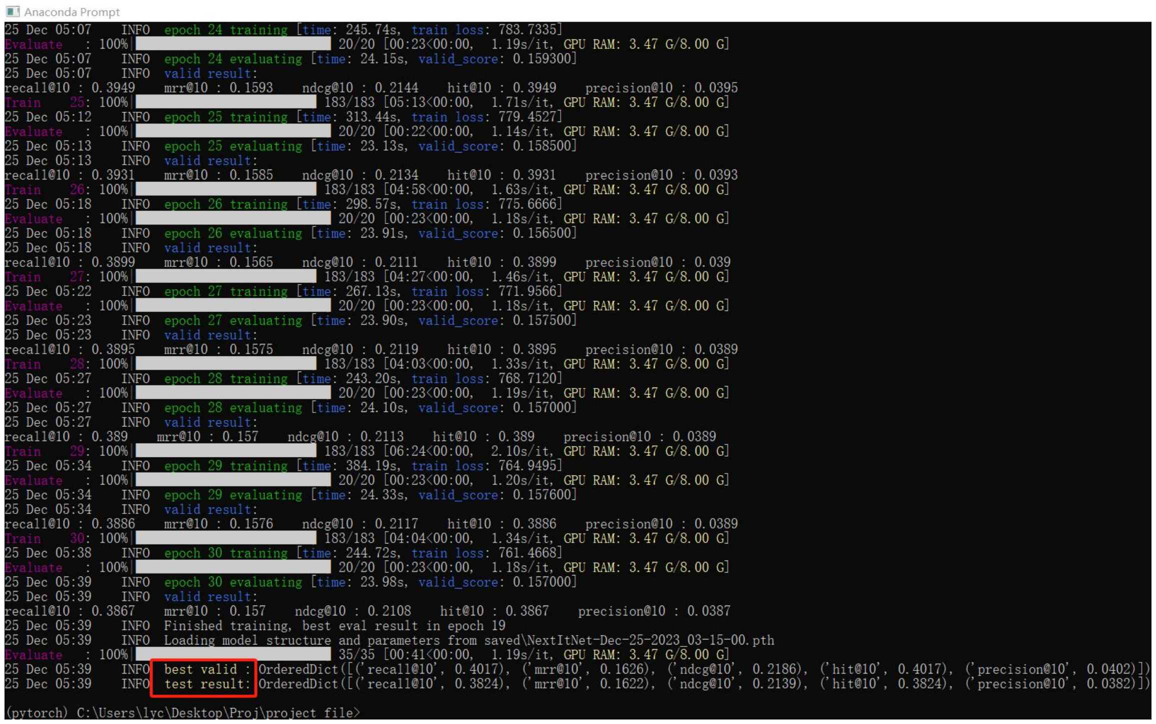


```
recall@10 : 0.1166    mrr@10 : 0.0375    ndcg@10 : 0.0557    hit@10 : 0.1166    precision@10 : 0.0117
Train    18: 100%|                              | 48/48 [01:04<00:00,  1.35s/it, GPU RAM: 1.79 G/8.00 G]
24 Dec 20:02    INFO  epoch 18 training [time: 64.93s, train loss: 255.7250]
Evaluate   : 100%|                                | 1/1 [00:00<00:00,  3.53it/s, GPU RAM: 1.79 G/8.00 G]
24 Dec 20:02    INFO  epoch 18 evaluating [time: 0.29s, valid_score: 0.036100]
24 Dec 20:02    INFO  valid result:
recall@10 : 0.1198    mrr@10 : 0.0361    ndcg@10 : 0.0553    hit@10 : 0.1198    precision@10 : 0.012
Train    19: 100%|                              | 48/48 [01:04<00:00,  1.35s/it, GPU RAM: 1.79 G/8.00 G]
24 Dec 20:03    INFO  epoch 19 training [time: 64.60s, train loss: 254.0615]
Evaluate   : 100%|                                | 1/1 [00:00<00:00,  3.51it/s, GPU RAM: 1.79 G/8.00 G]
24 Dec 20:03    INFO  epoch 19 evaluating [time: 0.29s, valid_score: 0.029300]
24 Dec 20:03    INFO  valid result:
recall@10 : 0.0944    mrr@10 : 0.0293    ndcg@10 : 0.0442    hit@10 : 0.0944    precision@10 : 0.0094
Train    20: 100%|                              | 48/48 [01:04<00:00,  1.35s/it, GPU RAM: 1.79 G/8.00 G]
24 Dec 20:04    INFO  epoch 20 training [time: 64.75s, train loss: 251.6409]
Evaluate   : 100%|                                | 1/1 [00:00<00:00,  3.49it/s, GPU RAM: 1.79 G/8.00 G]
24 Dec 20:04    INFO  epoch 20 evaluating [time: 0.29s, valid_score: 0.036500]
24 Dec 20:04    INFO  valid result:
recall@10 : 0.1071    mrr@10 : 0.0365    ndcg@10 : 0.0527    hit@10 : 0.1071    precision@10 : 0.0107
24 Dec 20:04    INFO  Finished training, best eval result in epoch 9
24 Dec 20:04    INFO  Loading model structure and parameters from saved\NextItNet-Dec-24-2023_19-42-24.pth
Evaluate   : 100%|                                | 1/1 [00:00<00:00,  3.56it/s, GPU RAM: 1.79 G/8.00 G]
24 Dec 20:04    INFO  best valid : OrderedDict([('recall@10', 0.123), ('mrr@10', 0.042), ('ndcg@10', 0.0605), ('hit@10', 0.123), ('precision@10', 0.0123)])
24 Dec 20:04    INFO  test result: OrderedDict([('recall@10', 0.1166), ('mrr@10', 0.0366), ('ndcg@10', 0.0548), ('hit@10', 0.1166), ('precision@10', 0.0117)])

(pytorch) C:\Users\lyc\Desktop\Proj\project file>
```

Figure 4: The running results of NextItNet on the built-in dataset

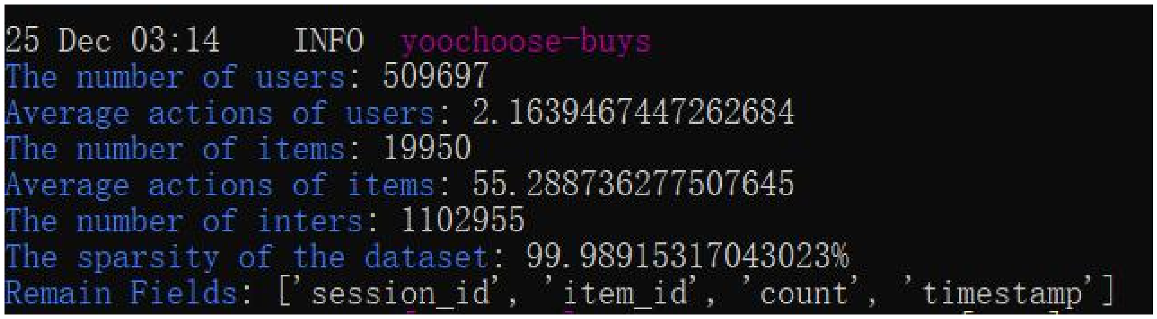
The results of using the yoochoose buy dataset show that the average training time per round is 5 minutes, and the training loss is about 770 seconds. The model performed the best on the validation set in the 19th round, recall@10 : 0.4017, mrr@10 : 0.1626, ndcg@10 : 0.2186, hit@10 : 0.4017, precision@10 : 0.0402. The result of the test set is recall@10 =0.3824, mrr@10 =0.1622, ndcg@10 =0.2139, hit@10 =0.3824, precision@10 =0.0382.(Figure 5 shows the training results, and Figure 6 shows the dataset parameters.)



```
Anaconda Prompt
25 Dec 05:07    INFO  epoch 24 training [time: 245.74s, train loss: 783.7335]
Evaluate   : 100%|                           | 20/20 [00:23<00:00,  1.19s/it, GPU RAM: 3.47 G/8.00 G]
25 Dec 05:07    INFO  epoch 24 evaluating [time: 24.15s, valid_score: 0.159300]
25 Dec 05:07    INFO  valid result:
recall@10 : 0.3949    mrr@10 : 0.1593    ndcg@10 : 0.2144    hit@10 : 0.3949    precision@10 : 0.0395
Train    25: 100%|                         | 183/183 [05:13<00:00,  1.71s/it, GPU RAM: 3.47 G/8.00 G]
25 Dec 05:12    INFO  epoch 25 training [time: 313.44s, train loss: 779.4527]
Evaluate   : 100%|                           | 20/20 [00:22<00:00,  1.14s/it, GPU RAM: 3.47 G/8.00 G]
25 Dec 05:13    INFO  epoch 25 evaluating [time: 23.13s, valid_score: 0.158500]
25 Dec 05:13    INFO  valid result:
recall@10 : 0.3931    mrr@10 : 0.1585    ndcg@10 : 0.2134    hit@10 : 0.3931    precision@10 : 0.0393
Train    26: 100%|                         | 183/183 [04:58<00:00,  1.63s/it, GPU RAM: 3.47 G/8.00 G]
25 Dec 05:18    INFO  epoch 26 training [time: 298.57s, train loss: 775.6666]
Evaluate   : 100%|                           | 20/20 [00:23<00:00,  1.18s/it, GPU RAM: 3.47 G/8.00 G]
25 Dec 05:18    INFO  epoch 26 evaluating [time: 23.91s, valid_score: 0.156500]
25 Dec 05:18    INFO  valid result:
recall@10 : 0.3899    mrr@10 : 0.1565    ndcg@10 : 0.2111    hit@10 : 0.3899    precision@10 : 0.039
Train    27: 100%|                         | 183/183 [04:27<00:00,  1.46s/it, GPU RAM: 3.47 G/8.00 G]
25 Dec 05:22    INFO  epoch 27 training [time: 267.13s, train loss: 771.9566]
Evaluate   : 100%|                           | 20/20 [00:23<00:00,  1.18s/it, GPU RAM: 3.47 G/8.00 G]
25 Dec 05:23    INFO  epoch 27 evaluating [time: 23.90s, valid_score: 0.157500]
25 Dec 05:23    INFO  valid result:
recall@10 : 0.3895    mrr@10 : 0.1575    ndcg@10 : 0.2119    hit@10 : 0.3895    precision@10 : 0.0389
Train    28: 100%|                         | 183/183 [04:03<00:00,  1.33s/it, GPU RAM: 3.47 G/8.00 G]
25 Dec 05:27    INFO  epoch 28 training [time: 243.20s, train loss: 768.7120]
Evaluate   : 100%|                           | 20/20 [00:23<00:00,  1.19s/it, GPU RAM: 3.47 G/8.00 G]
25 Dec 05:27    INFO  epoch 28 evaluating [time: 24.10s, valid_score: 0.157000]
25 Dec 05:27    INFO  valid result:
recall@10 : 0.389    mrr@10 : 0.157    ndcg@10 : 0.2113    hit@10 : 0.389    precision@10 : 0.0389
Train    29: 100%|                         | 183/183 [06:24<00:00,  2.10s/it, GPU RAM: 3.47 G/8.00 G]
25 Dec 05:34    INFO  epoch 29 training [time: 384.19s, train loss: 764.9495]
Evaluate   : 100%|                           | 20/20 [00:23<00:00,  1.20s/it, GPU RAM: 3.47 G/8.00 G]
25 Dec 05:34    INFO  epoch 29 evaluating [time: 24.33s, valid_score: 0.157600]
25 Dec 05:34    INFO  valid result:
recall@10 : 0.3886    mrr@10 : 0.1576    ndcg@10 : 0.2117    hit@10 : 0.3886    precision@10 : 0.0389
Train    30: 100%|                         | 183/183 [04:04<00:00,  1.34s/it, GPU RAM: 3.47 G/8.00 G]
25 Dec 05:38    INFO  epoch 30 training [time: 244.72s, train loss: 761.4668]
Evaluate   : 100%|                           | 20/20 [00:23<00:00,  1.18s/it, GPU RAM: 3.47 G/8.00 G]
25 Dec 05:39    INFO  epoch 30 evaluating [time: 23.98s, valid_score: 0.157000]
25 Dec 05:39    INFO  valid result:
recall@10 : 0.3867    mrr@10 : 0.157    ndcg@10 : 0.2108    hit@10 : 0.3867    precision@10 : 0.0387
25 Dec 05:39    INFO  Finished training, best eval result in epoch 19
25 Dec 05:39    INFO  Loading model structure and parameters from saved\NextItNet-Dec-25-2023_03-15-00.pth
Evaluate   : 100%|                           | 35/35 [00:41<00:00,  1.19s/it, GPU RAM: 3.47 G/8.00 G]
25 Dec 05:39    INFO  best valid : OrderedDict([('recall@10', 0.4017), ('mrr@10', 0.1626), ('ndcg@10', 0.2186), ('hit@10', 0.4017), ('precision@10', 0.0402)])
25 Dec 05:39    INFO  test result: OrderedDict([('recall@10', 0.3824), ('mrr@10', 0.1622), ('ndcg@10', 0.2139), ('hit@10', 0.3824), ('precision@10', 0.0382)])

(pytorch) C:\Users\lyc\Desktop\Proj\project file>
```

Figure 5: The running results of NextItNet on the yoochoose-buys



```
25 Dec 03:14    INFO  yoochoose-buys
The number of users: 509697
Average actions of users: 2.1639467447262684
The number of items: 19950
Average actions of items: 55.288736277507645
The number of inters: 1102955
The sparsity of the dataset: 99.98915317043023%
Remain Fields: ['session_id', 'item_id', 'count', 'timestamp']
```

Figure 6: Yoochoose-buy dataset parameters

4 Project Management

4.1 Activities

Objectives	**Tasks**
Collection of relevant literature	Read at least 20 articles in relevant fields, select and take notes
Project Proposal	Complete the proposal with clear structure and logic, including a reference list
Understanding models and mathematical methods	Analyze and compare the differences between models and interpret relevant methods
Improved model	According to the shortcomings of the existing model, search for possible improved technologies and methods through the literature
Experimental data processing	According to relevant research fields, the data set was searched and preprocessed
Experiment and Test	The model was implemented by code and applied to the dataset, trial and error
Summary	Analyze and summarize the experimental results to reach a conclusion, and complete the remaining writing
Paper Modify	Revise the format and improve the article
Presentation Prepare	Prepare PPT and review research work

Table 1: Activities

4.2 Schedule

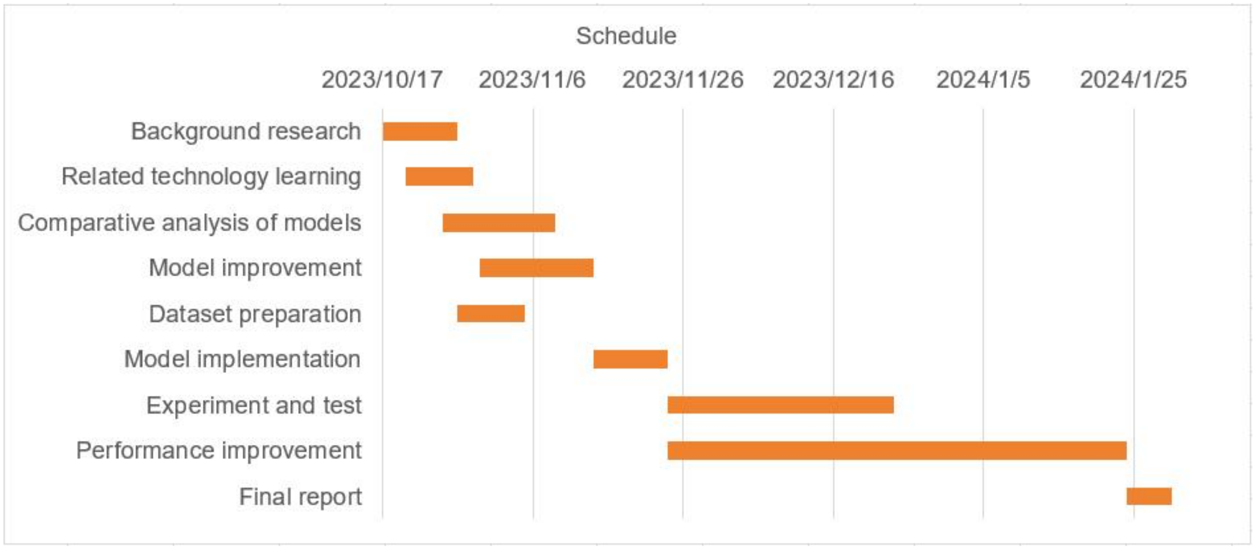


Figure 7: Gantt chart of thesis plan

4.3 Project Version Management

Use the Git repository and Feishu to manage project code or multiple versions of models that have been developed. The code will continue to be updated on Github's personal homepage, and backup management will also be carried out every time it is submitted through Flybook to facilitate subsequent changes.

4.4 Project Data Management

This article uses Zotero to manage relevant literature, including storing the literature in the Zotero manager and annotating and taking notes on each literature, making it easy to quickly search for the desired literature. For code files and dataset files, the code files will be uploaded to a personal account through Github for backup, and each version and date will be classified. Finally, regarding report file management, literature will be stored in cloud documents through the use of WPS office.

Literature management	Zotero
Code management	Github
Report management	WPS office

Table 2: Management

4.5 Project Deliverables

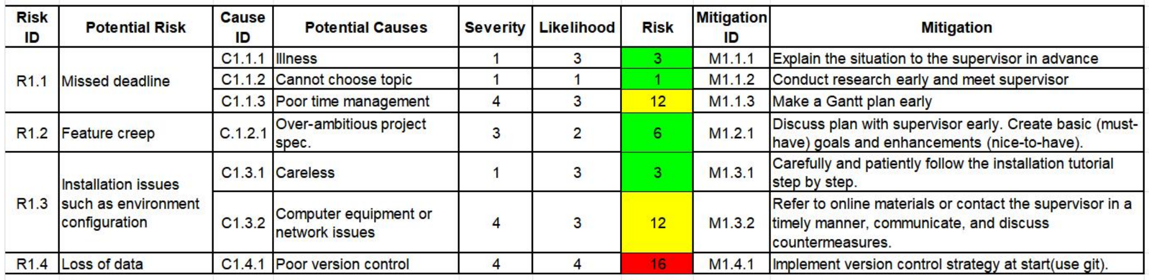
Project Resources	Deadline
weekly reports	Every week from the fourth week of the first semester
Project proposal, ethic form, plagiarism report	November 3, 2023
Progress Report	December 22, 2023
final report	April 12, 2024
presentation poster, video, and demo	May 29th to 31st, 2024
Tech Show	June 6th -7th, 2024

Table 3: Resources and Submission Date

5 Professional Issues and Risk

5.1 Risk Analysis

In the previous progress, there was a risk of missing the deadline, which occurred on December 18, 2023. Due to tight schedules in other disciplines, students reported to the supervisor in a unified manner. The supervisor decided to postpone the deadline for the progress report to December 27. This risk was resolved by reporting and communicating with the supervisor in advance. Another risk is environmental configuration and tool installation issues. I encountered many problems while installing Rebole, including downloading errors in the PyTorch version, mismatch between the PyTorch version and the Cuda version, incompatibility between PyTorch and the Cuda and Python versions, and very slow installation speed of Rebole. The first few problems were solved by searching a large amount of information on websites such as CSDN and spending a lot of time, The latter's network and device issues are resolved under the patient guidance of the supervisor. For future risks that may involve data loss, regular use of the Git tool for backup will be adopted.



Risk ID	Potential Risk	Cause ID	Potential Causes	Severity	Likelihood	Risk	Mitigation ID	Mitigation
R1.1	Missed deadline	C1.1.1	Illness	1	3	3	M1.1.1	Explain the situation to the supervisor in advance
		C1.1.2	Cannot choose topic	1	1	1	M1.1.2	Conduct research early and meet supervisor
		C1.1.3	Poor time management	4	3	12	M1.1.3	Make a Gantt plan early
R1.2	Feature creep	C.1.2.1	Over-ambitious project spec.	3	2	6	M1.2.1	Discuss plan with supervisor early. Create basic (must-have) goals and enhancements (nice-to-have).
R1.3	Installation issues such as environment configuration	C1.3.1	Careless	1	3	3	M1.3.1	Carefully and patiently follow the installation tutorial step by step.
		C1.3.2	Computer equipment or network issues	4	3	12	M1.3.2	Refer to online materials or contact the supervisor in a timely manner, communicate, and discuss countermeasures.
R1.4	Loss of data	C1.4.1	Poor version control	4	4	16	M1.4.1	Implement version control strategy at start(use git).

Table 4: Resources and Submission Date

5.2 Professional Issues

5.2.1 Legal issues

Privacy regulations: The recommendation system used this time will use datasets and involve the processing of user data, so it is necessary to ensure compliance with privacy regulations such as GDPR (European General Data Protection Regulations) or similar regulations in other countries/regions, and BCS's Code of Computer Conduct emphasizes the principles of protecting privacy and legitimate use of information.

Intellectual Property: This project uses publicly available datasets, algorithms, and code from others, and it is necessary to ensure compliance with relevant intellectual property regulations, such as ACM's Code of Ethics, which involves the principles of legitimate use of computing resources and respect for privacy.

5.2.2 Social issues

Social impact: My recommendation system may affect the sales of merchants, as recommended products are more likely to be purchased. This may lead to some small businesses losing competitiveness due to a lack of exposure, while large businesses will benefit more. Referring to the ACM Code of Professional Ethics, it is necessary to be aware of the broad impact of the computing profession on society and consider the long-term interests of society when designing recommendation systems. Ensure that the design of the system does not cause undue harm to specific individuals or groups.

5.2.3 Ethical issues

Transparency and interpretability: BCS's Code of Computer Conduct involves ethical principles, including transparency and respect for user choices. The model of a recommendation system is usually not publicly available, and it is necessary to consider how to improve transparency and interpretability so that users can understand why they receive specific recommendations.

5.2.4 Environmental issues

Energy consumption: My model may require a large amount of computing resources and may have negative impacts on the environment. BCS's Code of Computer Conduct emphasizes the responsibility of using resources, including energy consumption. Therefore, while considering the performance of the recommended model, resource consumption and cost issues should also be considered.

6 References

[1] F. Yuan, A. Karatzoglou, I. Arapakis, J. M. Jose, and X. He, ‘A Simple Convolutional Generative Network for Next Item Recommendation’. arXiv, Nov. 28, 2018. Accessed: Oct. 27, 2023. [Online]. Available: http://arxiv.org/abs/1808.05163

[2] C. Cheng, H. Yang, M. R. Lyu, and I. King, ‘Where You Like to Go Next: Successive Point-of-Interest Recommendation’.

[3] X. He and T.-S. Chua, ‘Neural Factorization Machines for Sparse Predictive Analytics’. arXiv, Aug. 16, 2017. Accessed: Oct. 27, 2023. [Online]. Available: http://arxiv.org/abs/1708.05027

[4] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, ‘Session-based Recommendations with Recurrent Neural Networks’. arXiv, Mar. 29, 2016. Accessed: Oct. 27, 2023. [Online]. Available: http://arxiv.org/abs/1511.06939

[5] J. Tang and K. Wang, ‘Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding’. arXiv, Sep. 19, 2018. Accessed: Oct. 27, 2023. [Online]. Available: http://arxiv.org/abs/1809.07426

[6] W. Song, S. Wang, Y. Wang, and S. Wang, ‘Next-item Recommendations in Short Sessions’. arXiv, Jul. 20, 2021. doi: 10.48550/arXiv.2107.07453.

[7] D. Chai, L. Wang, K. Chen, and Q. Yang, ‘Secure Federated Matrix Factorization’, *IEEE Intell. Syst.*, vol. 36, no. 5, pp. 11–20, Sep. 2021, doi: 10.1109/MIS.2020.3014880.

[8] Z. Li, G. Long, and T. Zhou, ‘Federated Recommendation with Additive Personalization’. arXiv, May 17, 2023. doi: 10.48550/arXiv.2301.09109.

[9] C. Kumar and M. Kumar, ‘Next-item recommendation within a short session using the combined features of horizontal and vertical convolutional neural network’, *Multimed. Tools Appl.*, 2023, doi: 10.1007/s11042-023-17201-z.