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

OXFORD BROOKES UNIVERSITY

UNDERGRADUATE PROJECT REPORT

Project Title:	**Book Recommendation Using Deep & Cross Network**
Surname:	**Cheng**
First Name:	**Arnold**
Student Number:	**201918020422**
Supervisor Name:	**Joojo Walker**
Module Code:	**CHC 6096**
Module Name:	**Project**
Date Submitted:	**May 5, 2023**

Declaration

Here, students would sign a statement indicating that they adhered to appropriate academic conduct in carrying out their final project.

Acknowledgment

I would like to thank module leader Dr. Joojo Walker for his teaching and advice, and also my supervisor for expert guidance and unwavering support.

Table of Contents

Declaration i
Acknowledgment ii
Table of Contents iii
Abstract iv
Abbreviations v
Glossary vii
Chapter 1 Introduction 1
1.1 Background 1
1.2 Aim 2
1.3 Objectives 2
1.4 Project Overview 2
1.4.1 Scope 3
1.4.2 Audience 3
Chapter 2 Background Review 4
Chapter 3 Methodology 7
3.1 Approach 7
3.1.1 Dataset 7
3.1.2 Model Conception 8
3.1.3 System Integration 13
3.2 Technology 13
3.3 Project Version Management 14
Chapter 4 Implementation and Result 15
4.1 Experimented Settings 15
4.1.1 Dataset Statistics 16
4.1.2 Evaluation Metrics 17
4.1.3 Baseline Model 18
4.1.4 Parameter Settings 19
4.2 Performance Comparison 20
4.3 Deployment Result 20
4.3.1 Ablation Study 20
4.3.2 Functionality Test 23

Chapter 5 Professional Issues 32
5.1 Project Management 32
5.1.1 Activities 32
5.1.2 Schedule 34
5.1.3 Project Data Management 36
5.1.4 Project Deliverables 37
5.2 Risk Analysis 38
5.3 Professional Issues 43
Chapter 6 Conclusion 46
References 47
Appendices 48

Abstract

This thesis aims to implement a book recommendation system and website based on DCN (Deep & Cross Network). The system analyzes and mines user behavior and preferences to recommend books that are in line with their interests. At the same time, the system also provides functions that bring users a better experience. In the era of the information explosion, how to find books quickly and accurately one is interested in has become a topic of concern for users. This thesis aims to design and implement a book recommendation system that allows users to easily obtain the book information they need. Currently, there are many book recommendation systems on the market. These systems mainly use user history and tag information to make recommendations, but the effect is not satisfactory. Therefore, this thesis adopts the DCN algorithm, which combines deep learning with the traditional CF algorithm, to improve the accuracy and efficiency of the recommendation system. The system also provides user account management and social sharing functions. The system improves the accuracy of the recommendation system. The system also uses a web framework to build the website, providing users with rich functions and a good user experience. Through testing, this thesis has implemented an efficient and accurate book recommendation system and website. The system recommends books that are in line with users' interests from their historical behavior and preferences and has good user experience and scalability.

Keywords: Recommendation System, Deep & Cross Network, Deep learning, Django

Abbreviations

1. DL: Deep Learning
2. DCN: Deep & Cross Network Model
3. RS: Recommendation System
4. NN: Neural Network
5. TR: Training Set
6. TS: Test Set
7. CV: Cross Validation
8. REG: Regularization
9. GD: Gradient Descent
10. LR: Learning Rate
11. WD: weight decay

Glossary

This section should have the definition of all the keywords you stated in the “Abstract” section. You can also define other relevant keywords. Particularly, if your final project report includes rare, unfamiliar, specialized, or made-up words or terms, the glossary serves as a dictionary for the reader to reference throughout their reading of the project report. (Note: this section should only contain definitions for specific terms in the project report. It does NOT function as an ordinary dictionary. Hence, common words related to the Computer Science and Software Engineering disciplines should NOT be included in this list.)

Chapter 1 Introduction

1.1 Background

The rapid development of web applications, especially mobile applications, has made it easy for people to browse a large number of web information resources, and the problem of information overload is a result of the rapid expansion of the Internet in terms of coverage and scale. There is so much information available at one time that it is difficult for users to filter it, reducing the effectiveness of information use[1], and how to recommend resources (e.g., goods, movies, books, etc.) that meet users' needs from the vast amount of information resources has become one of the current concerns of researchers.

A particularly promising approach to solving the information overload problem is the use of recommendation systems (RS), which are important information filtering tools. In 1997, Resnick & Varian [2]gave the now widely cited definition of a recommendation system: "It is the use of e-commerce websites to provide customers with product information and recommendations to help users decide what products they should buy, simulating a salesperson helping customers through the buying process". Extracting appropriate information from a large amount of information and recommending it to users can solve the problem of information overload and create a smooth and comfortable network environment[3].

Customer needs are often ambiguous, so if merchants can recommend products that meet users' ambiguous needs, they can convert users' potential needs into real needs and thus achieve increased product sales. And for different users, recommendation systems are committed to providing a recommendation system with different results for everyone[4]. Amazon's book recommendations, Apple Music's recommendations, and Taobao's product movie news usage recommendations all use these methods and have achieved significant benefits[5].

Recommender systems are emerging as a new research hotspot and also face the data sparsity problem (too small number of user ratings for recommended items) and the cold start problem (no rating data for newly recommended items and new users). Deep Learning (DL), a machine learning algorithm with recognition, analysis, and computation, brings new opportunities to alleviate data sparsity and cold start problems.

According to Singhal A et al., most of deep learning methods are enhancing collaborative filtering methods and are a significant improvement over matrix decomposition methods[6].

Therefore, the research on book recommendation systems incorporating deep learning is profoundly promising, with the goal of taking book recommendation systems to a whole new level of performance, reducing the difficulty of accessing information, and increasing the effectiveness of information delivery today. In my next work, I will use deep & crossing network (DCN) deep learning models to build a recommendation system that can recommend books more effectively. The rest of the report is structured as follows.

1.2 **Aim**

The aim of my project is to build and train a book recommendation system using the deep learning model Deep & Cross Network (DCN), Furthermore, a web application will be deployed with the deep learning model to recommend books.

1.3 **Objectives**

My objectives are as follows:

A). Research and review, study, and research of the book recommender system.

B). Completes the learning and research of deep learning and explores the book recommendation system.

C). Collects appropriate data for analysis and evaluation.

D). Finish data processing with property methods.

E). Design and implementation of the model and web, using python, Django, HTML, CSS, and JavaScript.

F). Test and evaluation of the model and the web application.

1.4 **Project Overview**

The purpose is to use a model of deep learning called Deep & Cross Network (DCN) [7]to train and build a book recommendation system. Because the DCN can effectively capture the interaction of limited effective features, learn highly nonlinear interactions, does not require manual feature engineering or traversal search, and have low computational cost, and also can more efficiently processing of data sparsity, and cold start problems of traditional recommendation systems.

The importance of my experiments is to use deep learning algorithms to learn and deal with complex problems like humans, to analyze and compute linear or nonlinear feature sequences from multiple dimensions in the face of complex scale data, to automatically learn features that match user needs from massive data, and to build recommendation systems.

1.4.1 **Scope**

The purpose is to use a model of deep learning called Deep & Cross Network (DCN) [7]to train and build a book recommendation system. Because the DCN can effectively capture the interaction of limited effective features, learn highly nonlinear interactions, does not require manual feature engineering or traversal search, and have low computational cost, and also can more efficiently processing of data sparsity, and cold start problems of traditional recommendation systems.

The importance of my experiments is to use deep learning algorithms to learn and deal with complex problems like humans, to analyze and compute linear or nonlinear feature sequences from multiple dimensions in the face of complex scale data, to automatically learn features that match user needs from massive data, and to build recommendation systems.

1.4.2 **Audience**

Users, merchants, and those who provide web services all benefit from this.

Because book recommendations solve the problem of users choosing a huge number of products quickly. And for a book site, generating more purchases is where the real money is. And recommending books that are more to the user's liking will undoubtedly increase user stickiness, improve user retention, and better attract users, and then the advertising revenue that comes with it will increase. This is also a win-win situation.

More efficient recommendations also save more reuse of resources, as well as a surge in search volume retrieval, maintaining the stable operation of the site.

The core needs of the Internet are growing, and recommendation systems are at the heart of that growth.

Chapter 2 Background Review

A recommendation system is a new research field that combines data mining, prediction algorithm [8], machine learning, and other disciplines. The first definition of a recommendation system was given in the literature[2], which pointed out that in daily life, whether it is known or unknown events, people need to make decisions at all times. In the face of familiar things, people can often rely on experience to make reasonable decisions. However, in the face of unknown things, people need other people's oral suggestions, book reviews, film reviews, recommendations, etc. to make judgments.

In literature[9], it is believed that the recommendation system is to match a large number of items for different users that meet their interests and preferences but are not observed by users. It is believed that the recommendation system is becoming an important business with significant economic impact. Essentially, recommendation system is a simulation of human behavior. It analyzes and processes specific data information through recommendation algorithm, and then recommends the processed results to users with relevant needs[10]. The recommendation algorithm is the core of the recommendation system. It can model according to the user's historical purchase needs, behavior records or similar preferences, to find the requirements that meet the user's preferences and recommend them to the user. The formal definition of recommendation system [11], [12] is as follows:

Let P represent the collection of all users, and C represent the collection of objects that users can recommend. In practical problems, P and C are very large collections. The function f indicates the preference of user p for c, that is, $f: P \times C \rightarrow R$, where R represents a finite sequence of non-negative real numbers, and the recommendation object $c' \in C$ that makes the function f obtains the maximum value is recommended to the user. As shown in formula:

$$\forall p \in P,\ c_p' = \arg\max_{c \in C} f(p,c) \tag{1}$$

c_p' represents the recommendation object that best meets the user's p preference.

Therefore, before selecting the most interesting object for users, the recommendation system must use known user recognition to complete the prediction of the unknown recommendation object recognition, which is the process of recommendation system extrapolation. In recent years, recommendation technologies have been classified from different perspectives, and different scholars have given different connotations to recommendation systems. At present, traditional recommendation systems are divided into three categories[13]: content-based recommendation (CB)[14], collaborative filtering recommendation (CF)[15], and hybrid recommendation[16].

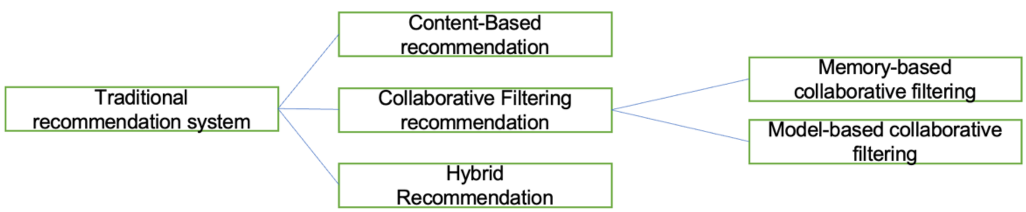

Figure1: traditional recommendation system

Recommendation system based on deep neural network.

Deep Neural Network (DNN) is one of the deep learning models[17]–[20], which can also be called multi-layer neural network or multi-layer perceptron (MLP). At present, the trend of introducing deep neural network technology into personalized recommendation is increasingly obvious [18], [21], [22].

In the essay [20], the deep neural network model was first integrated into the field of video recommendation, and simulation experiments were conducted on YouTube video website. YouTube video website is characterized by many registered users, fast video updates, different video duration, and a large number of videos. It is difficult for traditional recommendation algorithms to recommend video content that meets users' preferences. The recommendation process is divided into two stages: candidate set generation and video sequencing. The candidate set generation stage can be regarded as a process of video filtering, that is, select a video set similar to the user's viewing history from the existing videos according to the user's viewing records as the next recommended candidate video. In the candidate set generation stage, the video recommendation problem is regarded as a multi-classification problem, and the user and video are modeled using a deep neural network. The simulation results show that the recommendation model proposed in the literature[20] has a high recall rate and efficiency and can train video data sets with a scale of millions However, the model still has the following shortcomings: 1) In the face of massive video data, the model only cleans the data simply; 2) Video websites often have malicious videos (such as advertisements).

A recommendation model combining DNN and matrix decomposition is proposed in reference[19], which can quickly establish the nonlinear model required for generating interaction functions for user items. Compared with the single matrix decomposition algorithm, the model further improves the accuracy of scoring prediction results and improves the recommendation performance; However, the model does not extract user preferences from multiple dimensions, and its generalization ability is poor. To solve this problem, a DNN-based deep hybrid recommendation model was proposed in[18]. The model inputs the user and project information into the improved machine learning model for training, and further studies the interaction between the user and the recommended project from multiple dimensions. The feature learning part of the model for users and projects is composed of two parallel DNNs. The purpose is to extract the potential features of static projects and dynamic users, to accurately predict user preferences and improve recommendation performance.

In [17], a Wide&Deep model is proposed to solve large-scale online recommendation problems. This model is a combination of a single-layer Wide part and a multi-layer Deep part.

The Wide&Deep model mainly uses the Wide part to learn the characteristics of target users, and the Deep part to generalize similar recommendation items. It can train 500 billion samples, effectively alleviate the problem of data sparsity, and can also be used for classification, regression, search, and other problems; Its shortcoming is that it needs artificial feature engineering.

Deep & cross network (DCN)[7] is to replace the wide part with the cross-layer network based on wide&deep to handle feature crossing.

Chapter 3 Methodology

3.1 **Approach**
3.1.1 **Dataset**

I will use the Amazon dataset in my model training and deploying.

The Amazon book dataset is a rich and valuable resource for researchers and developers in the fields of e-commerce, natural language processing, data mining, and machine learning. It contains comprehensive information, reviews, and ratings for millions of books sold on the Amazon website, making it an excellent source for training and testing various machine learning models, including product recommendations, sentiment analysis, and text classification.

The dataset includes essential details for each book, such as the title, ISBN, author, publisher, publication date, page count, price, sales rank, book description, editor recommendation, customer ratings, and reviews. Additionally, the Amazon Reviews dataset provides a significant amount of book-related review data that can be used for tasks such as sentiment analysis and topic modeling.

By utilizing this dataset, researchers and developers can build various prediction models, such as predicting which books will become bestsellers or which books may appeal to users. Furthermore, these data can provide insights into trends in different markets, enabling businesses to make informed decisions regarding their products and services.

3.1.2 **Model Conception**

A brief description of the Deep & Cross Network model

The Deep & Cross Network model we will refer to as the DCN model below:

In this project, the DCN (Deep & Cross Network) model is employed as the core recommendation model for the proposed solution. The complete DCN v1 model is shown in the figure:

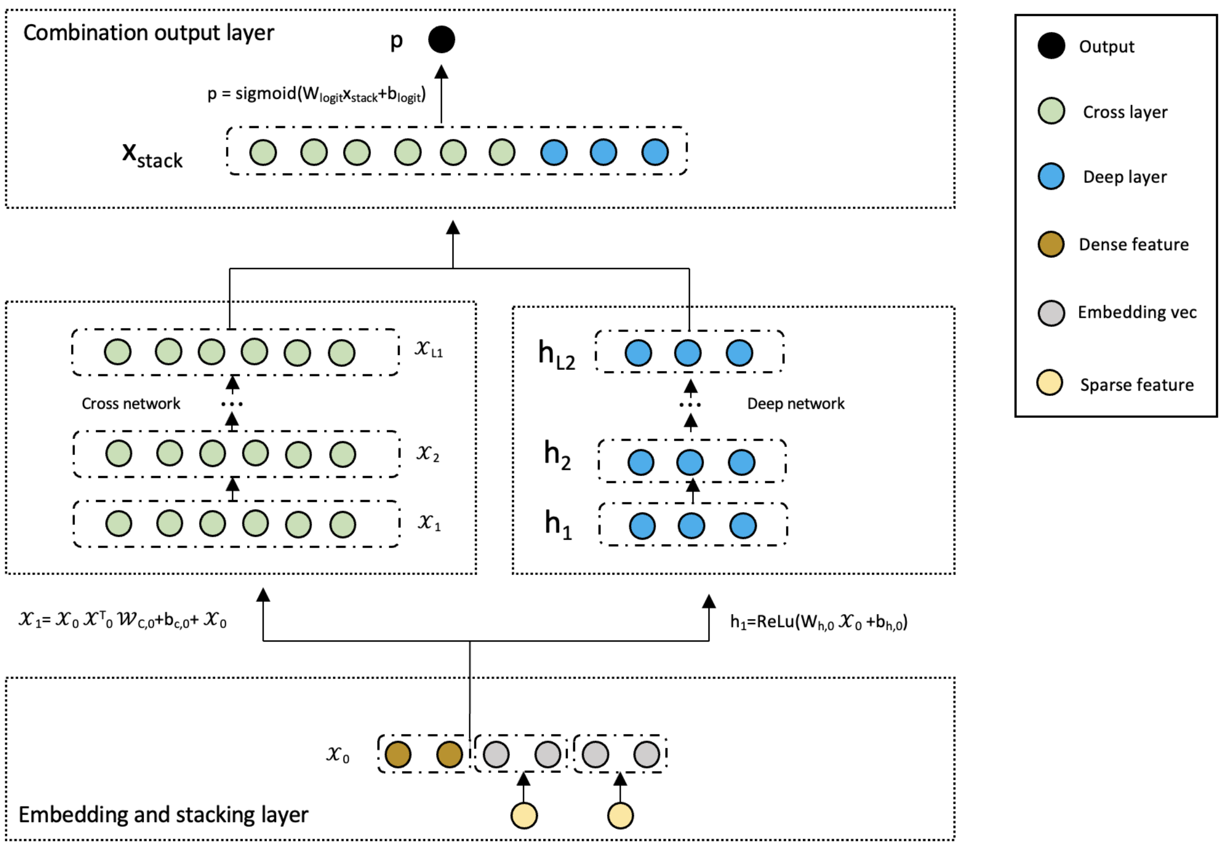

Figure 2: DCN v1 Model

DCN can be divided into four main parts. The first part, located at the bottom, is the "Embedding and Stacking Layer". This layer involves the process of converting discrete features into dense vectors using an embedding technique, and then stacking these embedding vectors with the continuous features to form a unified input vector for the subsequent layers.

When dealing with input data that exhibit both discrete and continuous characteristics, such as in network-scale recommendation systems like click-through rate (CTR) prediction, the input is typically comprised of categorical features, such as "country=USA". These categorical features are frequently represented using one-hot encoding, resulting in a binary vector such as "[0,1,0]". However, this approach often results in a high-dimensional feature space when dealing with large vocabularies.

To reduce the dimensionality, the embedding process is used to convert these discrete features into a dense vector of real values (often called an embedding vector):

$$x_{embed,i} = W_{embed,i} x_i \tag{2}$$

Then, superimpose the embedding vector with the continuous eigenvector to form a vector:

$$x_0 = \left[x_{embed,1}^T \;\cdots\cdots\cdots\; x_{embed,k}^T, x_{dense}^T\right] \tag{3}$$

The stitched vector x_0 will serve as input to our Cross Network and Deep Network

The second and third parts are the "Cross Network" and the "Deep Network", respectively, which are in the middle of the architecture. The Cross Network utilizes explicit feature interactions to capture the pairwise correlations between different features, while the Deep Network leverages the power of deep neural networks to learn hierarchical representations of the input data.

The core idea of cross network is to apply explicit feature intersections in an efficient way. A cross network consists of intersecting layers, each with the following formula:

$$x_{l+1} = x_0 x_l^T w_l + b_l + x_l = f(w_l, b_l, x_l) + x_l \tag{4}$$

A few parameters of the cross-network limit the model capacity. To capture highly nonlinear interactions, the model introduces a deep network in parallel.

A deep network is a fully connected feedforward neural network, and each depth layer has the following formula:

$$h_{l+1} = f(W_l h_l + b_l) \tag{5}$$

Finally, the top part of DCN is the "Combination Output Layer", which combines the outputs from the Cross Network and Deep Network to produce the final prediction result. This layer plays a crucial role in integrating the complementary strengths of the two networks and improving the overall performance of the model.

The link layer connects the outputs of the two parallel networks and passes through a full link layer to get the output:

$$\rho = \sigma([x_{L1}^T, x_{L2}^T]w_{logits}) \tag{6}$$

The explanation of DCN v2 model calculation principle.

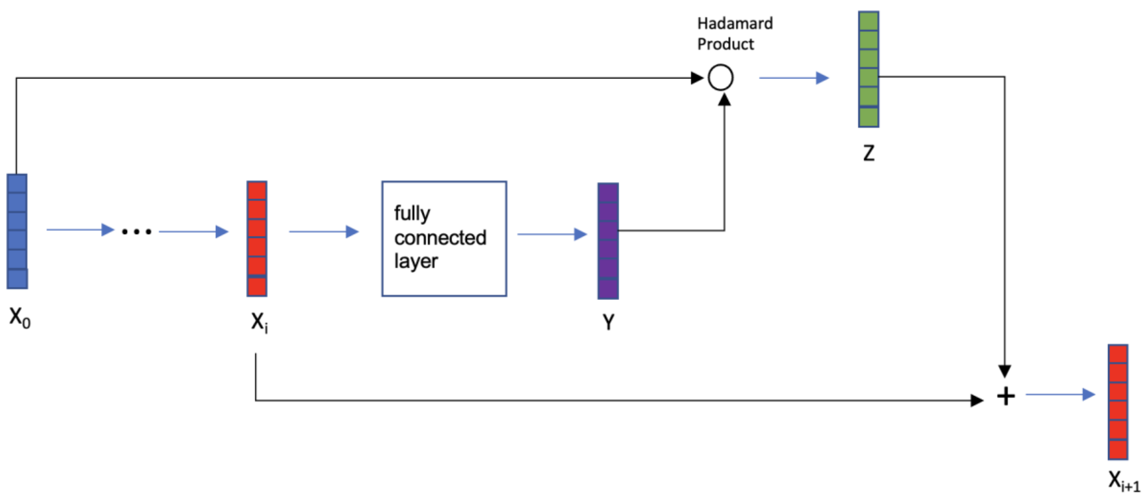

Figure 3: Cross-layer

The cross-layer is the fundamental building block of a cross-network, and the figure3 is the structure of a single cross-layer. Given an input vector $\mathcal{X}_0$ that passes through i neural network layers, the output vector is $\mathcal{X}_i$. Taking the i-th cross layer as an example, the vector $\mathcal{X}_i$ is fed into a fully connected layer, which generates another vector Y. The lowest-level vector $\mathcal{X}_0$ is then subjected to Hadamard product with vector Y to produce the output vector Z. Vectors $\mathcal{X}_i$ and Z represent input and output, respectively, and the sum of these two vectors yields $\mathcal{X}_{i+1}$ (similarly to the skip connections in ResNet). Vector $\mathcal{X}_{i+1}$ represents the output of the i-th cross layer, while vectors $\mathcal{X}_0$ and $\mathcal{X}_i$ are inputs to this layer, and all parameters are contained within the fully connected layer.

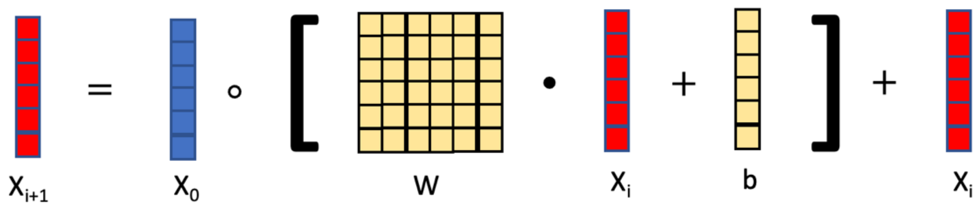

Figure 4: formula of Cross-layer

The cross-layer can be represented by the equation in figure 4. The input of the cross-layer consists of two vectors, $\mathcal{X}_0$ and $\mathcal{X}_i$. Here, $\mathcal{X}_0$ represents the input at the lowest layer of the neural network, while $\mathcal{X}_i$ represents the input to the i-th layer of the neural network. The square bracketed section denotes a fully connected layer, which computes the product of matrix W and vector $\mathcal{X}_i$ and adds vector b. The output of the fully connected layer is a vector with the same size as the input vector $\mathcal{X}$. The matrix W and the vector b are the parameters of the fully connected layer, which need to be updated during training using gradient descent. Finally, the element-wise multiplication (Hadamard product) between the vector $\mathcal{X}_0$ and the output of the fully connected layer is taken, followed by addition with vector $\mathcal{X}_i$. The resulting vector is denoted as $\mathcal{X}_{i+1}$, which serves as the output of the cross-layer. Notably, both the input and output are vectors with the same shape.

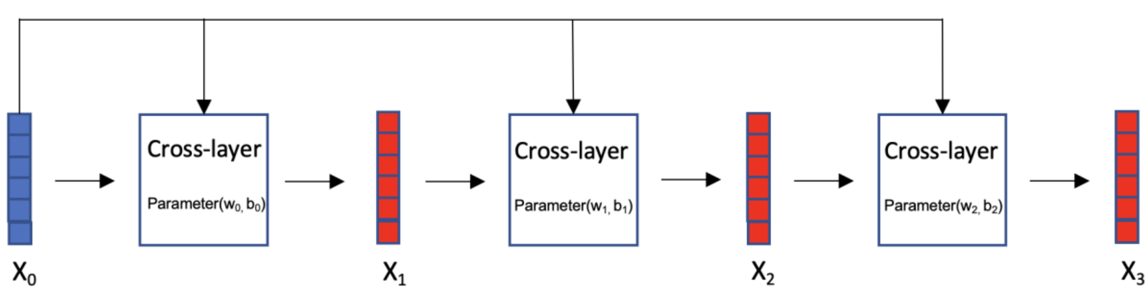

Figure 5: Cross network

Cross network, as figure 5: Vector $\mathcal{X}_0$ is the input of the cross-network, which is fed into a cross-layer with parameters W_0 and b_0, resulting in output vector $\mathcal{X}_1$. Then, $\mathcal{X}_1$ is fed into the next cross-layer, along with $\mathcal{X}_0$, both of which serve as inputs to this cross-layer. The parameters for this cross-layer are W_1 and b_1, and the resulting output vector is $\mathcal{X}_2$. This process is repeated iteratively.

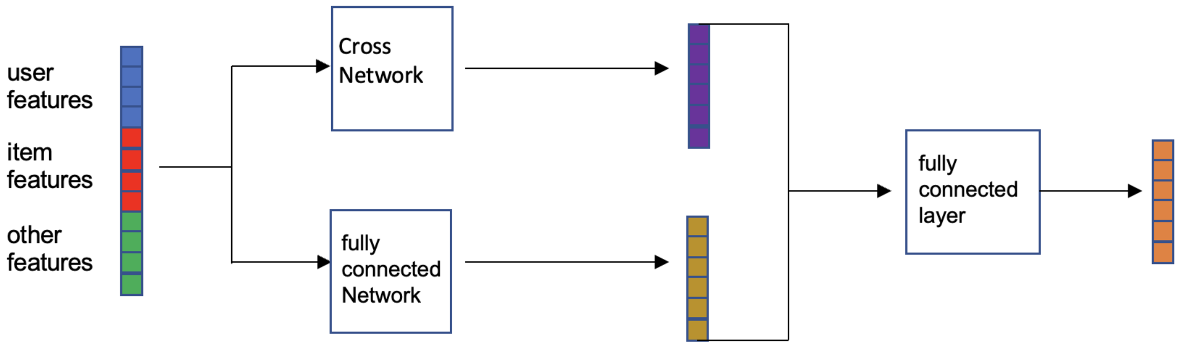

Figure 6: Deep & Cross Network

Deep cross network (DCN) is a combination of cross network and fully connected network. In recommendation systems with ranking models, the input includes user features, item features, and other features, which are concatenated and fed into two neural networks in parallel: a fully connected network and a cross-network. Each neural network outputs a vector, which is then concatenated and fed into a fully connected layer that outputs a vector. The concatenation of the fully connected network, cross-network, and fully connected layer is the DCN. The DCN can be used for both recall and ranking.

3.1.3 **System Integration**

This is a system architecture diagram designed to visually demonstrate the application of the model on a web page.

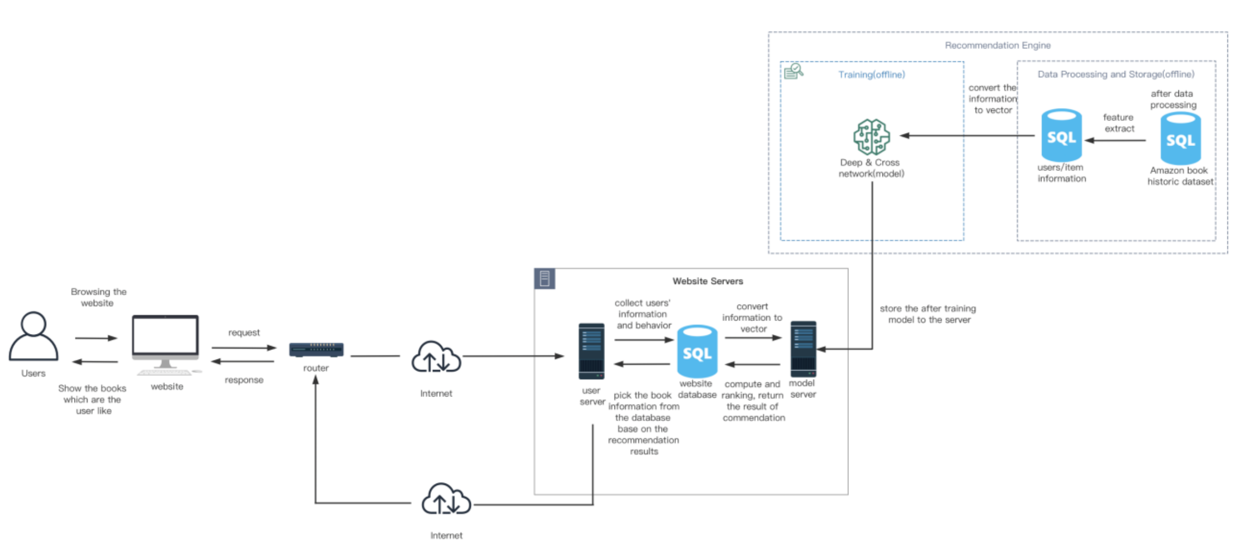

Figure 7: Architecture system diagram

3.2 **Technology**

These are the hardware and software that I'm going to use in the project.

HARDWARE	**SOFTWARE**
MACBOOK PRO 16 INCHES	Colab
SOC: M1 PRO	Pytorch environment
GPU: A5000(RENT FROM AUTODL)	Google Drive
	Language: python

	Jupyter notebook Autodl

Table1： tools

3.3 **Project Version Management**

I use both local storage and Google Cloud to manage my projects, and the papers and data are stored locally before being uploaded to a folder in the cloud and synced. When the codes have been done. It will automatically upload the code to Google Cloud after moving it to the cloud folders. Here is the link: https://github.com/Arnold201918020422/book-recommendation-system.git

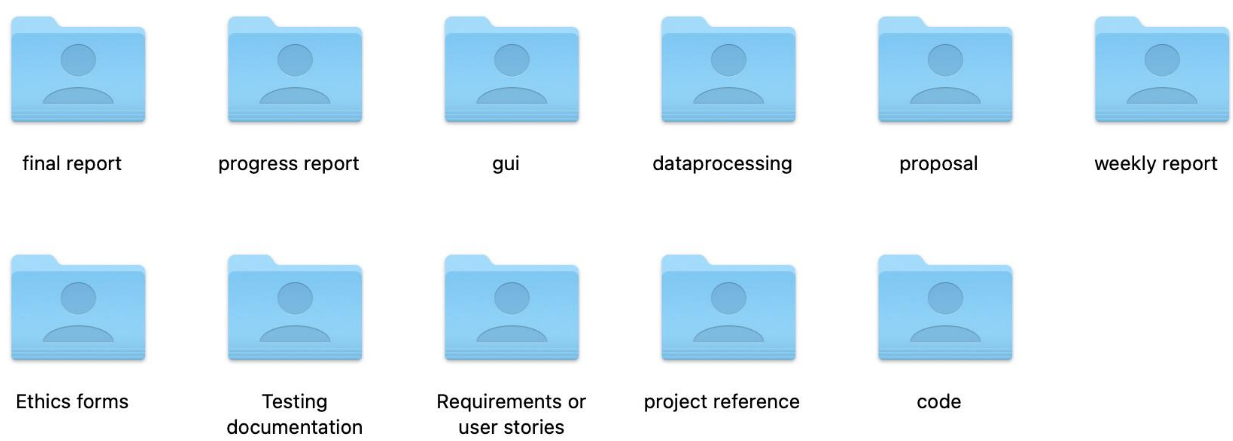

Figure 8: management folders

Chapter 4 Implementation and Result

4.1 Experimented Settings

This model is a Deep & Cross Network (DCN) used for binary classification tasks. It consists of the following parts:

Feature processing: For sparse features in the input data, an embedding layer is used to transform them into fixed-size low-dimensional vectors; for dense features, they are directly used as inputs to the model.

Deep neural network (DNN): The dense and low-dimensional sparse features processed above are concatenated and fed through a multi-layer fully connected neural network to extract high-level feature representations.

Cross network (CrossNet): Through cross-feature engineering, different features interact with each other to improve the model's non-linear capability.

Model fusion: The outputs of DNN and CrossNet are concatenated, and then another layer of fully connected neural network is added to output probabilities between 0 and 1, which represent the probability of being positive samples.

The hyperparameters include:

Name	Meaning
feat_size	the number of features
embedding_size:	the dimension of dense features
linear_feature_columns:	list of dense features
dnn_feature_columns:	list of sparse features
cross_num:	A parameterization method for the cross network
cross_param:	number of layers in the cross network
dnn_hidden_units:	number of hidden neurons in the DNN hidden layers
init_std:	standard deviation for parameter initialization
l2_reg:	L2 regularization coefficient

drop_rate:	dropout probability in the DNN

Table 2: notaion

In data preprocessing, sparse features and dense features are stored separately in different lists, and LabelEncoder and MinMaxScaler are used to encode and scale them. Then, these feature columns are combined into the required format for the DCN model.s

Next, the data is split into training and testing sets, converted to TensorDataset format required by PyTorch, and train_loader and test_loader are created to batch-read the training and testing data.

The DCN model consists of two parts: Cross Network and Deep Network. The Cross Network is responsible for learning cross features to capture high-order features, while the Deep Network is responsible for learning non-linear relationships between features. The model uses BCELoss as the loss function, Adam optimizer for parameter updates, and outputs training loss and testing AUC after each epoch.

4.1.1 **Dataset Statistics**

'reviews_Books_5.json.gz'—3.2GB	'meta_Books.json.gz'—1.2GB
The 'reviews_Books_5.json.gz' dataset is a collection of book reviews from Amazon's Book category. It contains customer reviews and ratings for books, along with information about the reviewer such as their ID, name, and helpfulness votes. The data is in JSON format and has been compressed using gzip.	The 'meta_Books.json.gz' dataset is a collection of metadata for books that are available on Amazon. It includes information such as the book title, author name, price, publisher, publication date, and genre/category.
The dataset was last updated in September 2018 and contains over 9 million reviews spanning from May 1996 to July 2014. It is commonly used in natural language processing (NLP) research and sentiment analysis tasks.	This dataset also contains additional information such as product descriptions, reviews/average ratings, sales rank, and similar products. The data is in JSON format and has been compressed using gzip. The dataset was last updated on September 2018 and contains information on over 22 million

	books. It is commonly used in natural language processing (NLP) research and recommender system tasks.

4.1.2 **Evaluation Metrics**

The following evaluation metrics are used in this code:

1.BCELoss: Binary Cross-entropy Loss function is used to measure the difference between the model's predicted output and the true label.

$$BCELoss=-\frac{1}{N}\sum_{i=1}^{N}[y_i\log(\hat{y}_i)+(1-y_i)\log(1-\hat{y}_i)] \qquad (7)$$

the above formula, NN represents the total number of samples, yiy_i represents the true label of the ii-th sample, and y^i$\hat{y}_i$ represents the predicted output of the model for the i-th sample. BCELoss is a binary cross-entropy loss function, which measures the degree of difference between the model's predicted output and the true label. In this code snippet, we use BCELoss to calculate the loss of the model during training and update the model parameters through backpropagation to gradually approach the optimal solution.

2.AUC: Area Under the ROC Curve is calculated using the get_auc function in the code to evaluate the classification performance of the model on the test set. AUC is the area under the ROC curve, which represents the probability that the classifier ranks a positive sample higher than a negative sample. It ranges from 0 to 1, and values closer to 1 indicate better classifier performance.

AUC is the area under the ROC curve, which can be used to evaluate the performance of a classifier. The ROC curve is plotted with false positive rate (FPR) on the x-axis and true positive rate (TPR) on the y-axis. Each point on the ROC curve represents the classification results of a classifier at different thresholds. The formula for calculating AUC is::

$$AUC=\int_0^1 TPR(FPR^{-1}(t))dt \qquad (8)$$

where TPR represents true positive rate, FPR represents false positive rate, FPR−1*FPR*−1 represents the inverse function of FPR, that is, find the minimum FPR value greater than or equal to a given t value among all false positive rates and return its corresponding TPR. This inverse function is used to convert FPR to FPR−1*FPR*−1, making it easy to calculate AUC.

In summary, AUC is the area under the ROC curve, representing the probability that the model predicts a positive sample ranked higher than a negative sample. It ranges from 0 to 1, with values closer to 1 indicating better model performance.

4.1.3 **Baseline Model**

Comparison of DCN model with other four models, which are:

1. LR model (Logistic Regression Model): This model takes all integer features and performs log mapping and discretization processing, then uses single and cross features as inputs, where cross features are selected using complex feature selection tools. The best hyperparameter search result includes 42 cross features.

LR model is relatively simple, easy to implement, and can handle sparse data well. However, it cannot capture complex nonlinear relationships between features.

2. FM model (Factorization Machine Model): This model is a common recommendation system model used to learn the relationship between users and items. It takes the original feature vector as input and models the interaction between each pair of features using matrix factorization techniques.

FM model can better capture interaction information between features, but when facing high-dimensional sparse data, the model parameters are too large and difficult to train.

3. DNN model (Deep Neural Network Model): This model is a standard feedforward neural network that can automatically extract high-level feature representations from raw features. The best hyperparameter search result includes a 5-layer network with each layer having a hidden vector dimension of 1024.

DNN model can automatically learn higher-level feature representations, but requires sufficient amounts of data and computing resources to train deep networks.

4. Wide&Deep model: This model combines wide linear models and deep neural networks to improve recommendation system performance. It utilizes domain knowledge to construct wide parts of cross-features but skips this part in comparisons due to a lack of good cross-feature selection methods.

Wide&Deep model combines linear models and deep neural networks, utilizing domain knowledge to construct wide parts of cross features, but may affect model performance due to lack of good feature selection methods.

5. DeepCrossing model: This model does not have explicit feature crossing, instead it uses stacking and residual units to build implicit feature crossing.

DeepCrossing model uses stacking and residual units to perform feature crossing, effectively preventing overfitting and improving model performance.

The best hyperparameter search result includes a 5-layer residual unit with input dimensions of 424 and hidden vector dimensions of 537.

6. DCN (Deep Cross Network): This model combines deep neural networks and cross networks to improve recommendation system performance. DCN model combines deep neural networks and cross networks, considering both low-order and high-order interactions of input features, and has better predictive performance and interpretability.

DCN: Best hyperparameters search result: 2-layer deep network, 6-layer cross-layer, deep network hidden vector of 1024 dimensions.

4.1.4 **Parameter Settings**

Optimization strategy:	Search space:
Adam optimizer is used with a batch size of 512	Hidden vector dimension of the deep network: 32~1024
Batch normalization is applied to the deep network	Number of layers in the deep network: 2~5
Gradients with norm exceeding 100 are clipped	Number of layers in the cross layer: 1~6
Regularization: Early stopping strategy is used as no evidence of L_2 regularization or dropout being effective was found in the paper.	Initial learning rate: 0.0001~0.001
Hyperparameters: Hidden vector dimension of the deep network, number of layers in the deep network, number of layers in the cross layer and initial learning rate are obtained through hyperparameter grid search.	

4.2 **Performance Comparison**

MODEL	DCN	DC	DNN	FM	LR
LOGLOSS	0.17	0.21	0.19	0.22	0.26

Table 8: Logloss

Logloss	**0.17**	**0.20**	**0.27**
DNN	3.2×10^7	1.7×10^7	6.7×10^6
DCN	7.8×10^6	5.9×10^6	3.7×10^4

Table 10: Number of parameters required to achieve the specified best validation set Logloss

When one is training a neural network, it is necessary to set the model's parameters or capacity, which includes factors such as the number of neurons in each layer and the activation functions used. A larger model capacity may be better suited to fit the training data but can lead to overfitting issues.

Thus, when designing a neural network, it is essential to strike a balance between the model's capacity and the loss function on the validation set. The "number of parameters required to achieve the specified best validation set Logloss" refers to the quantity of model parameters needed to achieve the best performance on a given validation set. This value is smaller for simpler models but may result in underfitting. Conversely, a larger value could indicate overfitting.

4.3 **Deployment Result**

4.3.1 **Ablation Study**

This section is to study the contribution of different components of your model toward the overall performance.

The experimental objects include:

batch_size

epochs

embedding_size

other parameters:

lr = 1e-2

wd = 1e-3

seed = 2022

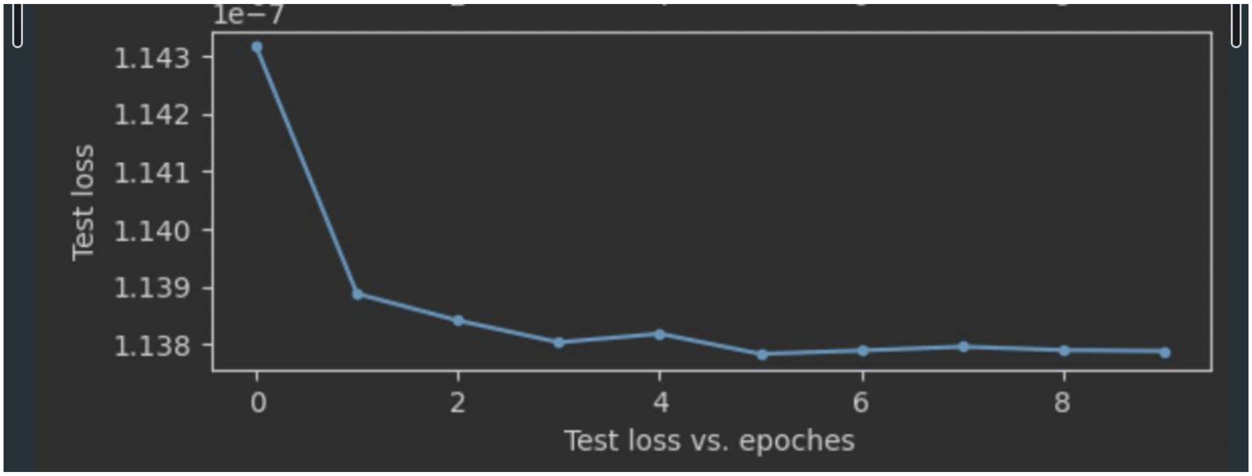

Figure 9: batch_size: 512 epochs: 10 embedding_size: 8

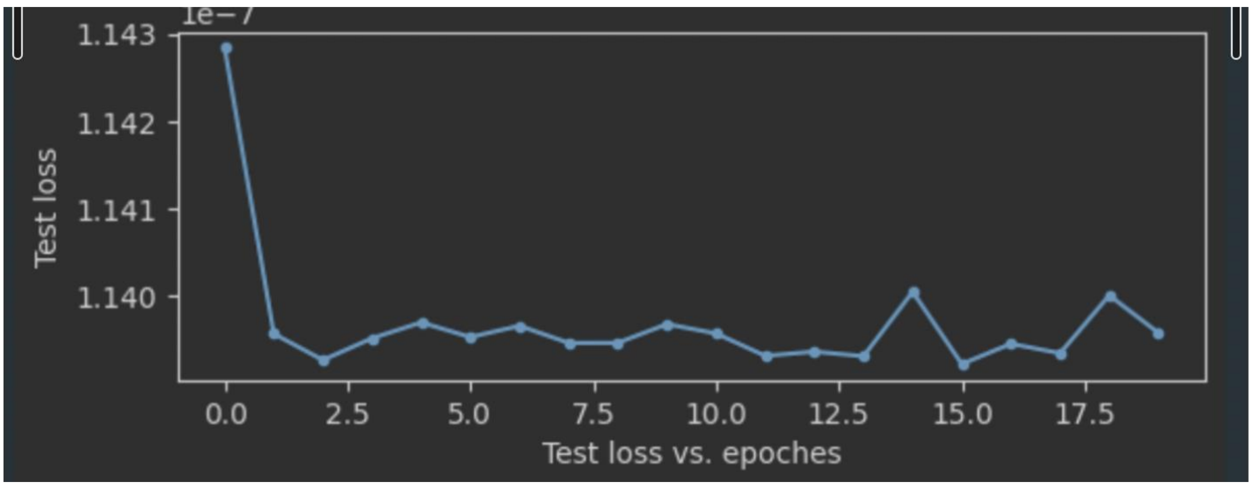

Figure 10: batch_size: 256 epochs: 20 embedding_size: 8

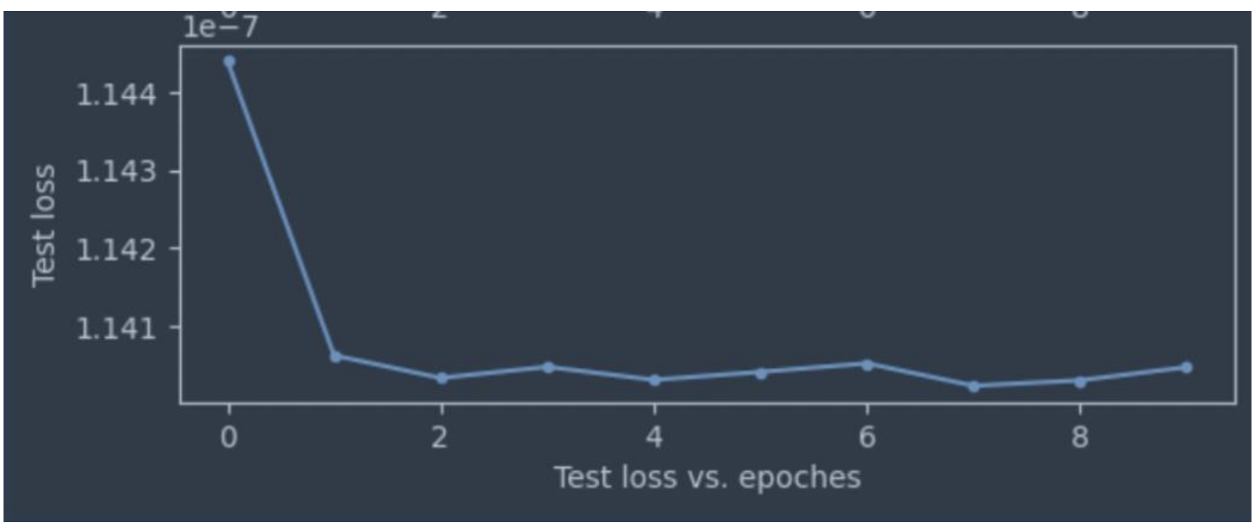

Figure 11: batch_size: 256 epochs: 10 embedding_size: 8

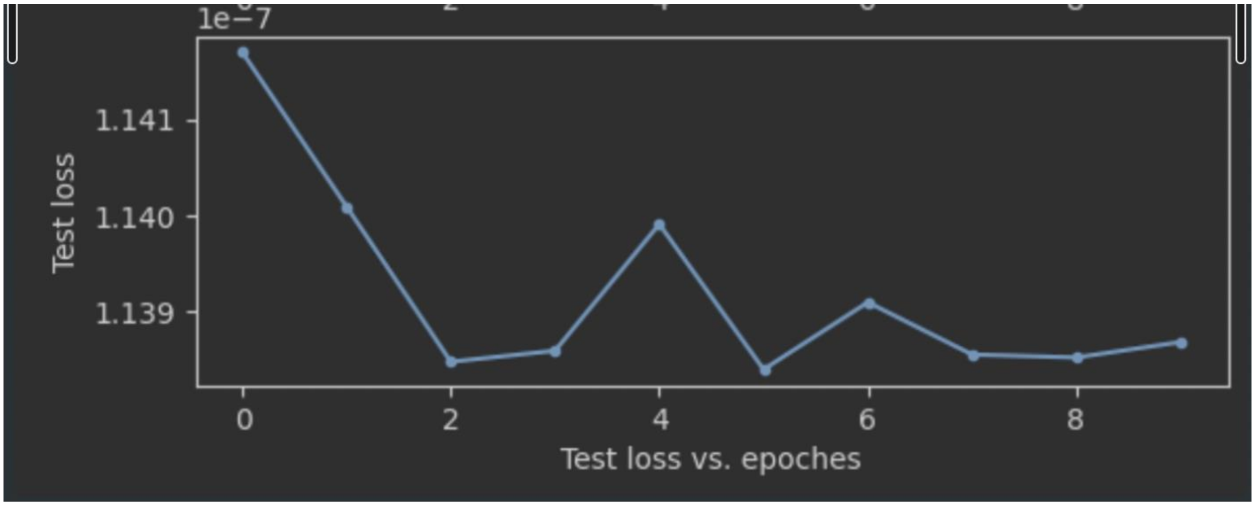

Figure 12: batch_size: 128 epochs: 10 embedding_size: 8

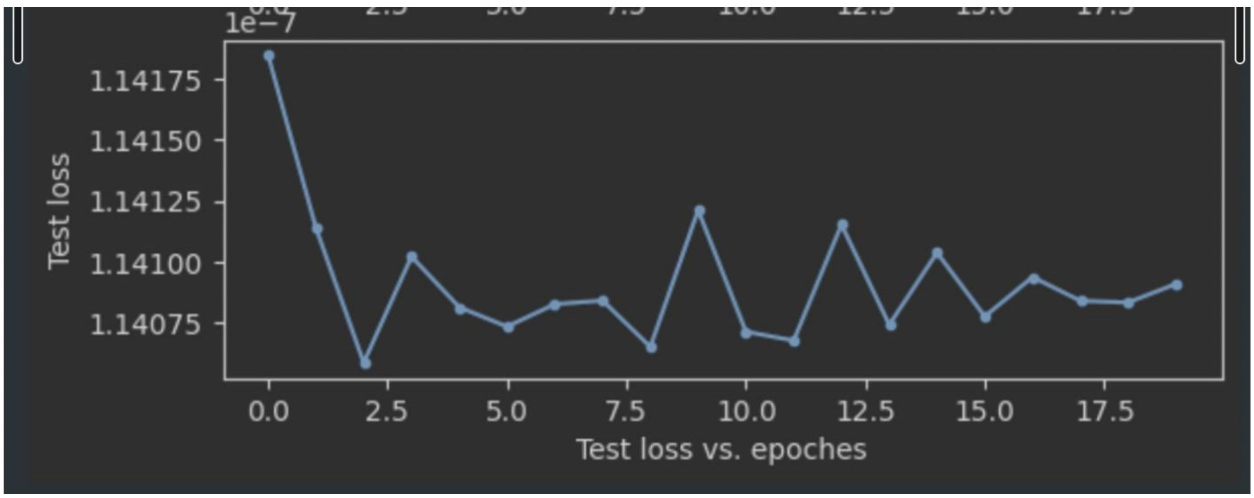

Figure 13: batch_size: 128 epochs: 20 embedding_size: 8

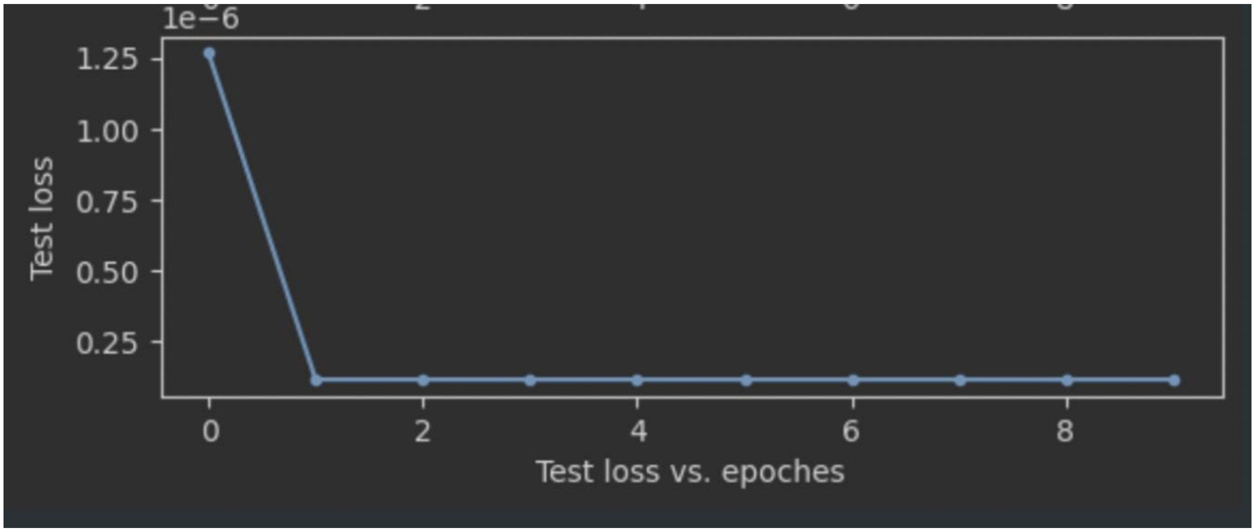

Figure 14: batch_size: 128 epochs: 20 embedding_size: 4

4.3.2 **Functionality Test**

Functional testing is a type of software testing that involves testing the system or software's functionality to verify if it meets specific requirements and standards. Prior to conducting functional testing, test cases and test data must be prepared. Test cases should include various inputs and expected outputs under different scenarios. Test data can be manually created or generated automatically.

The following are the steps involved in functional testing:

Determine test cases.

Run the software.

Verify if the software output meets the expected results.

Record test results and issues.

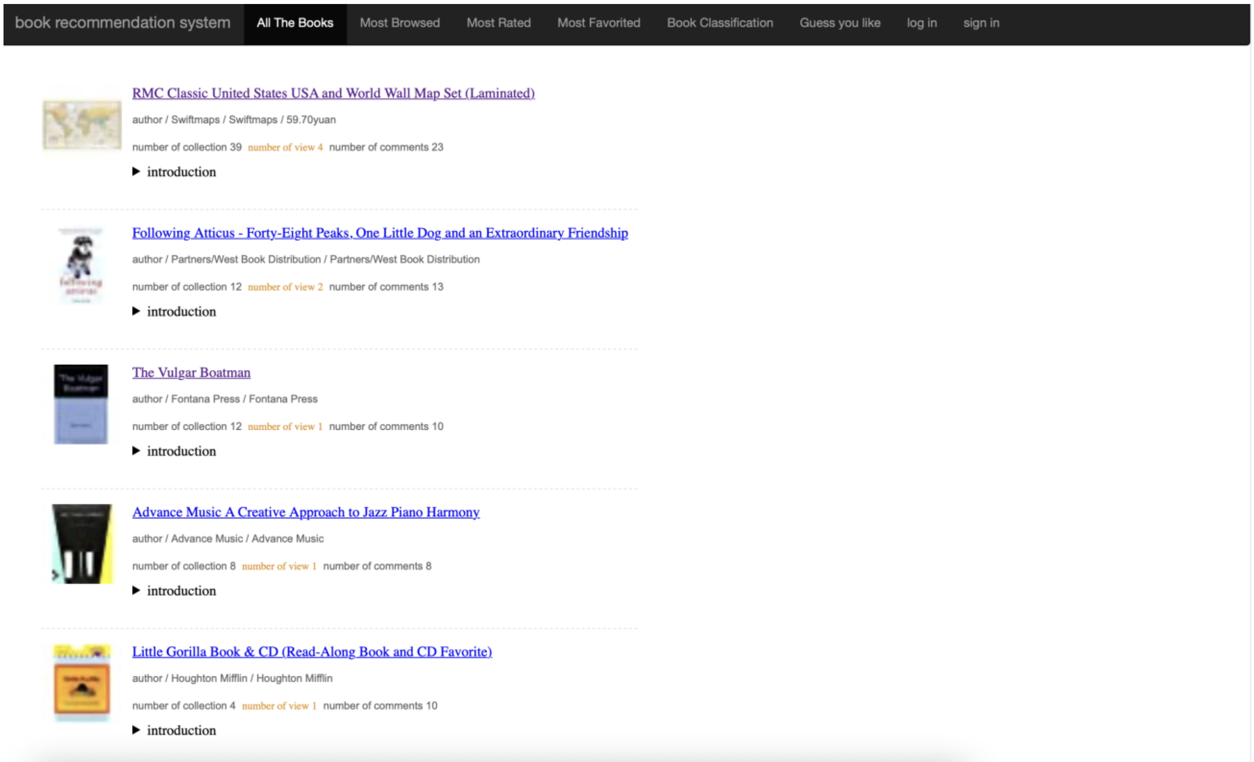

Figure 15: user interface

Function: click, log in, sign in, user management

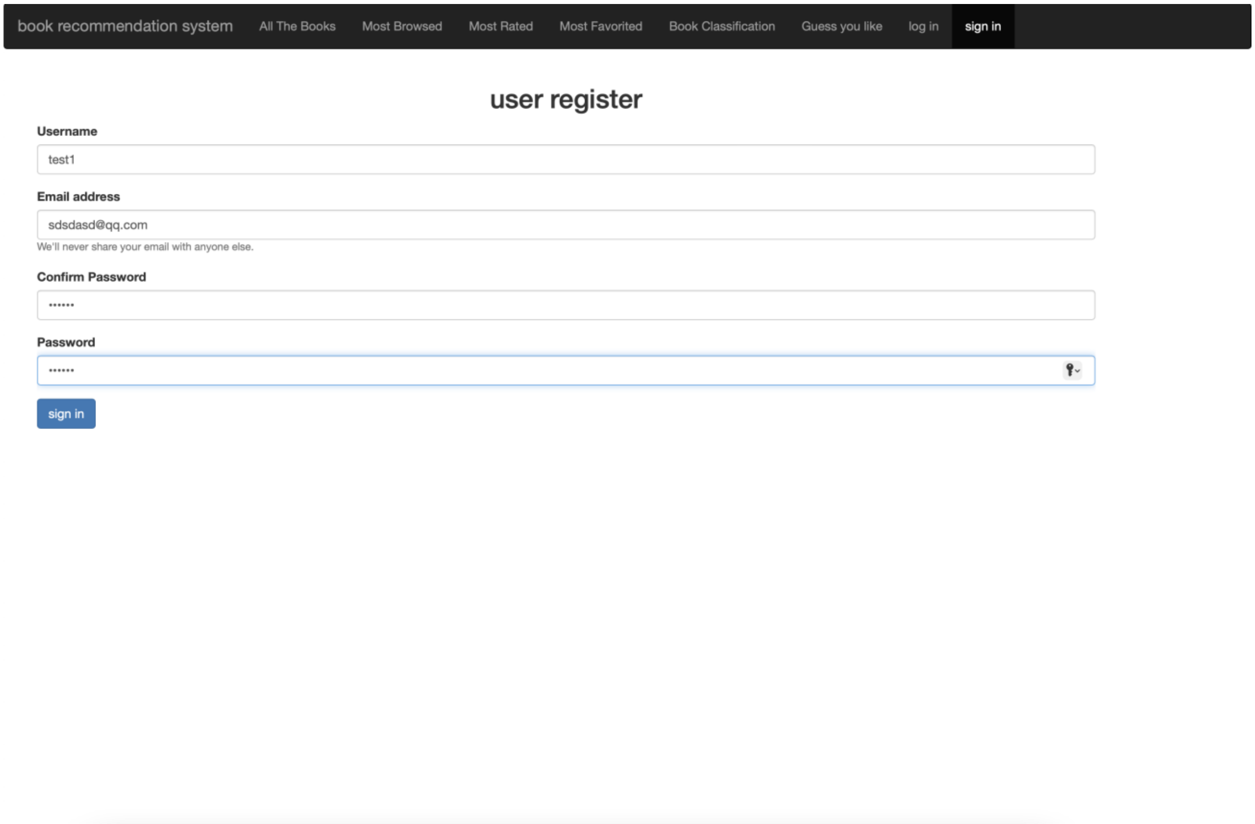

Figure 16: sign in function

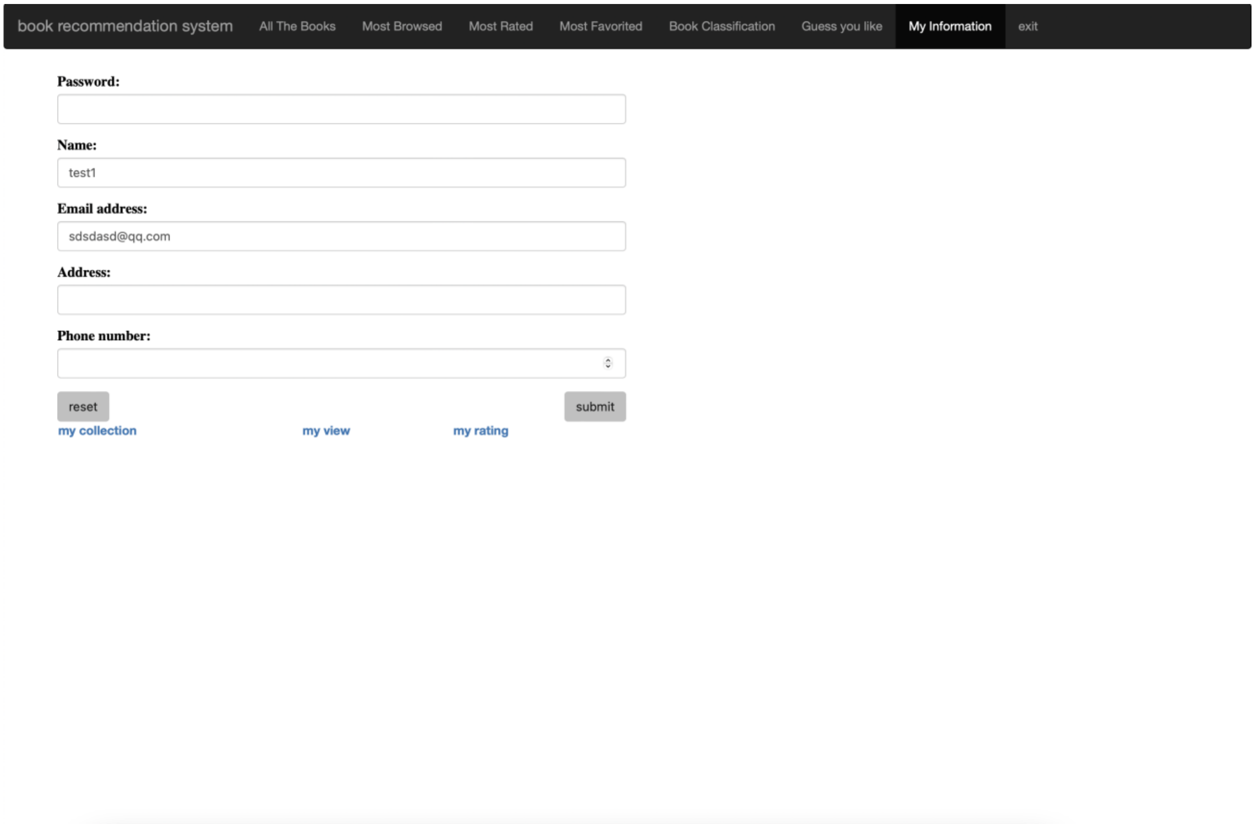

Figure 17: my information

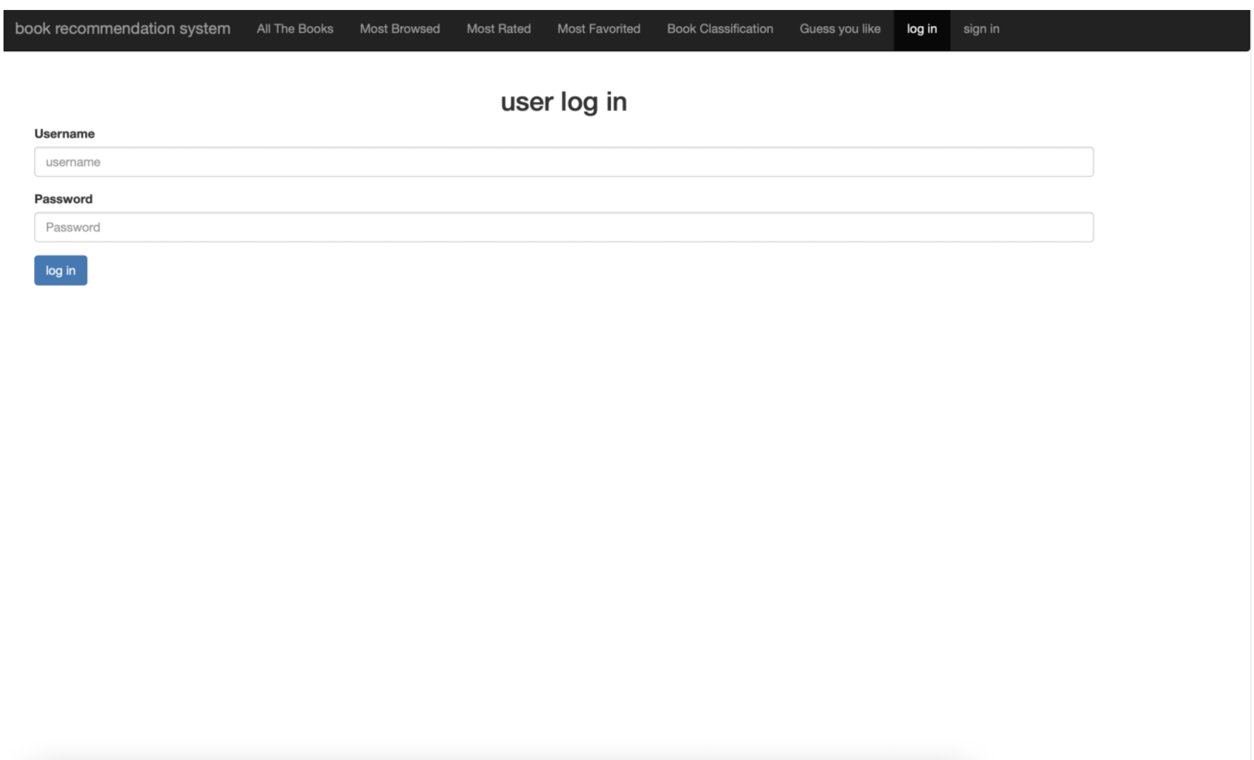

Figure 18: login function

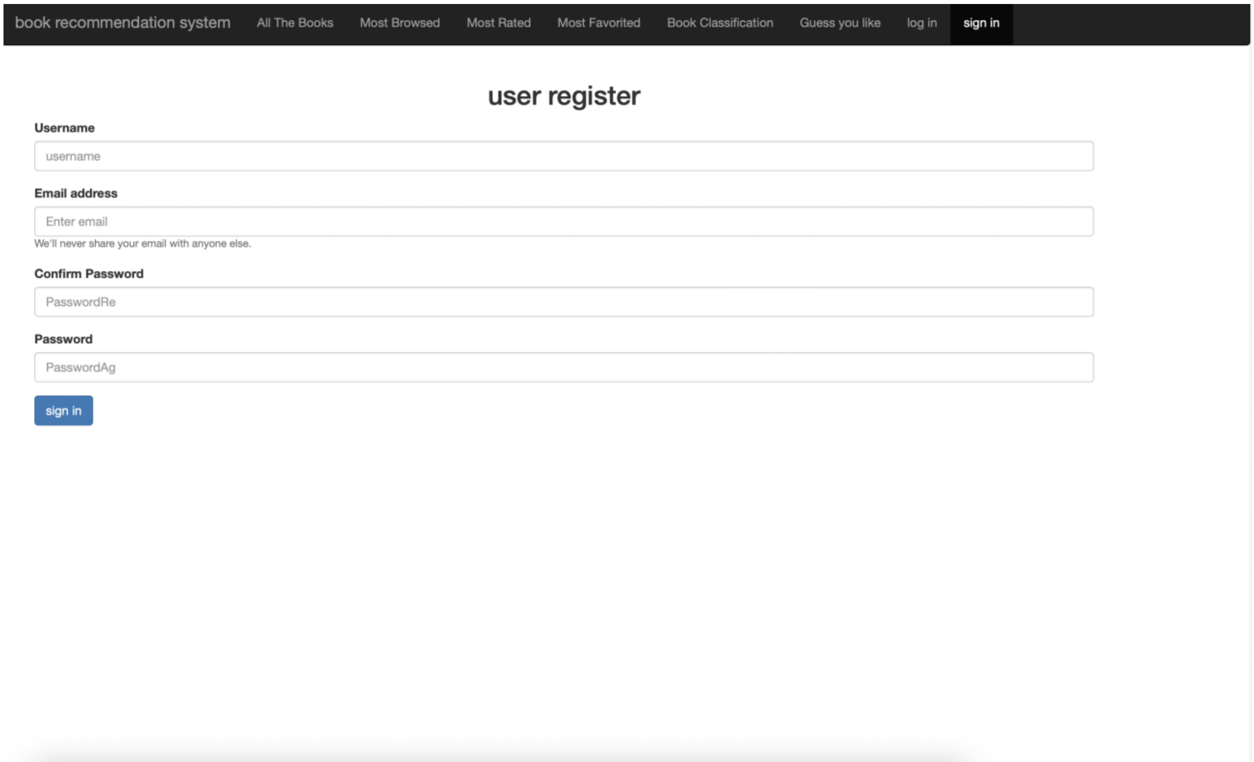

Figure 19: register interface

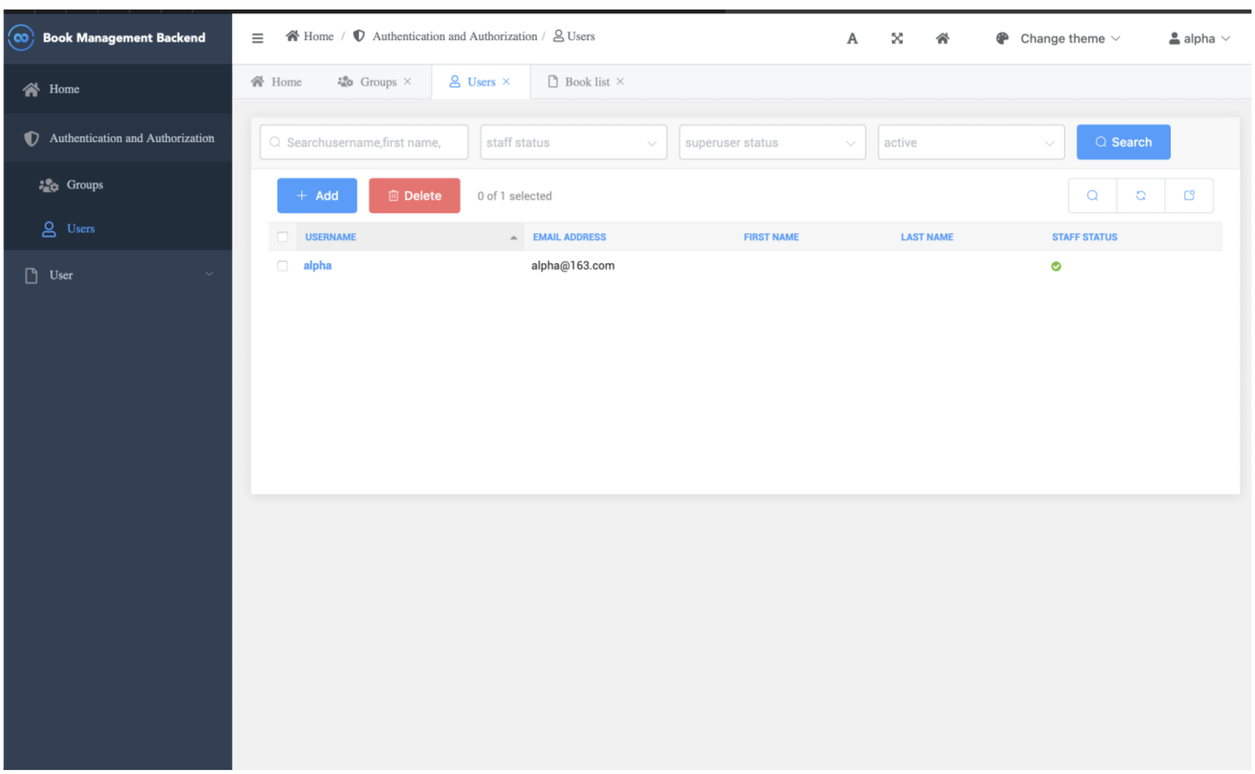

Figure 20: management interface

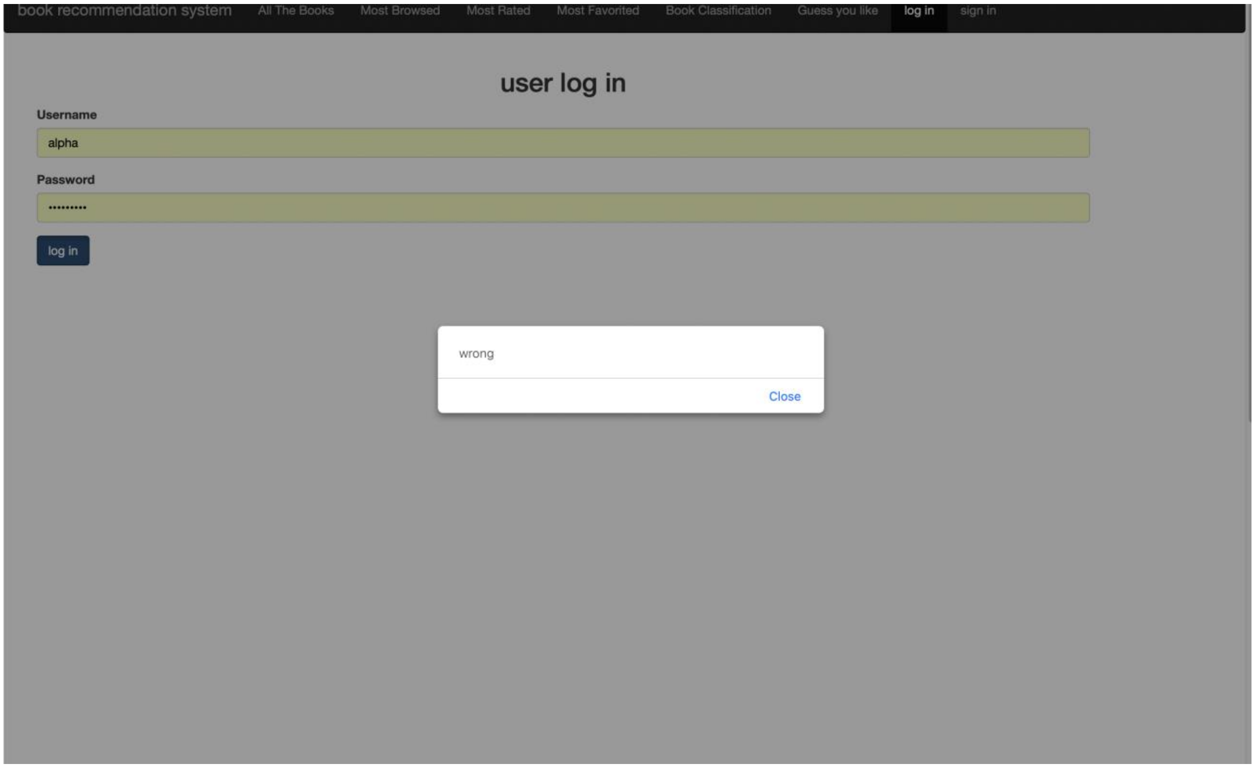

Figure 21: sign in with wrong password

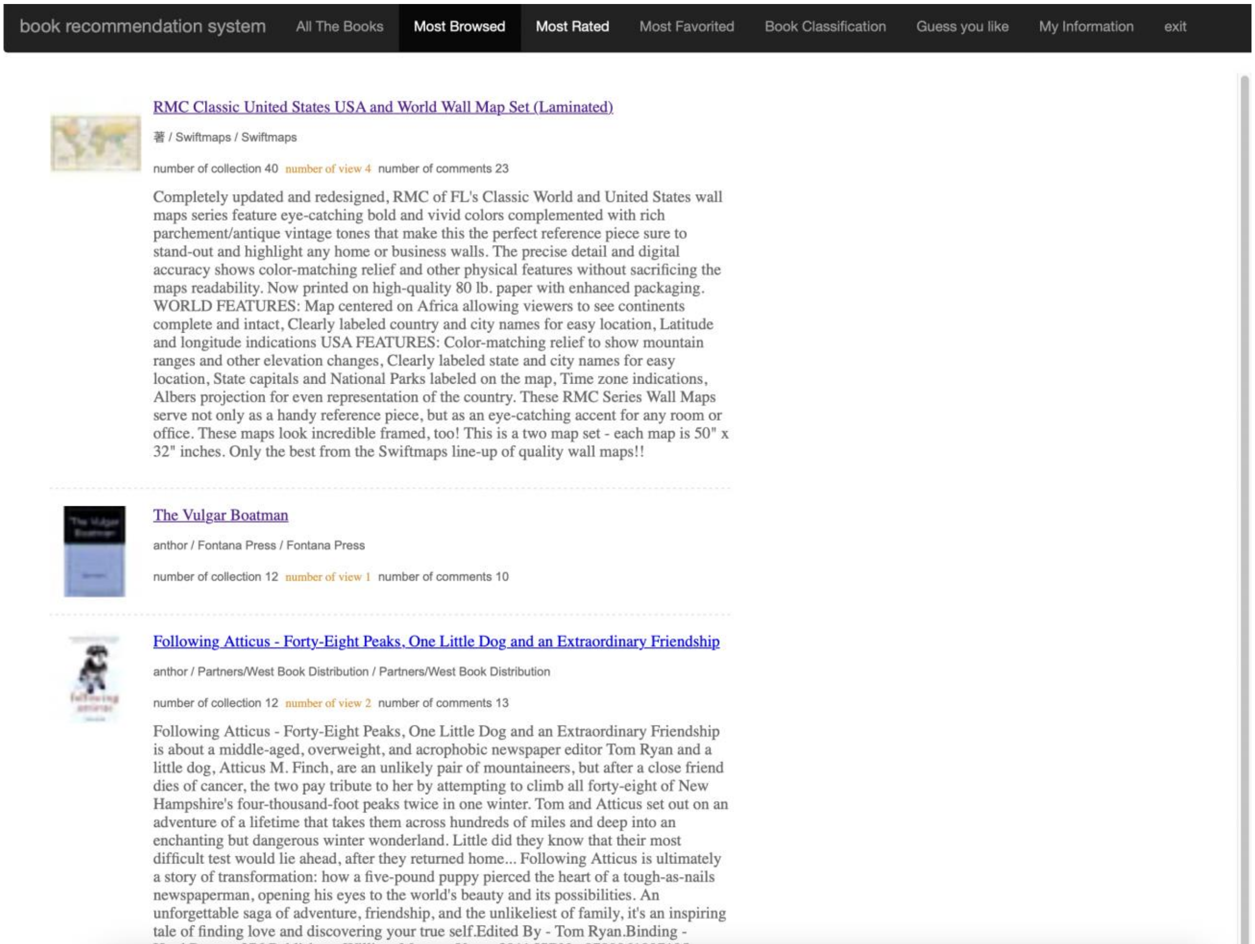

Figure 22: most browsed function

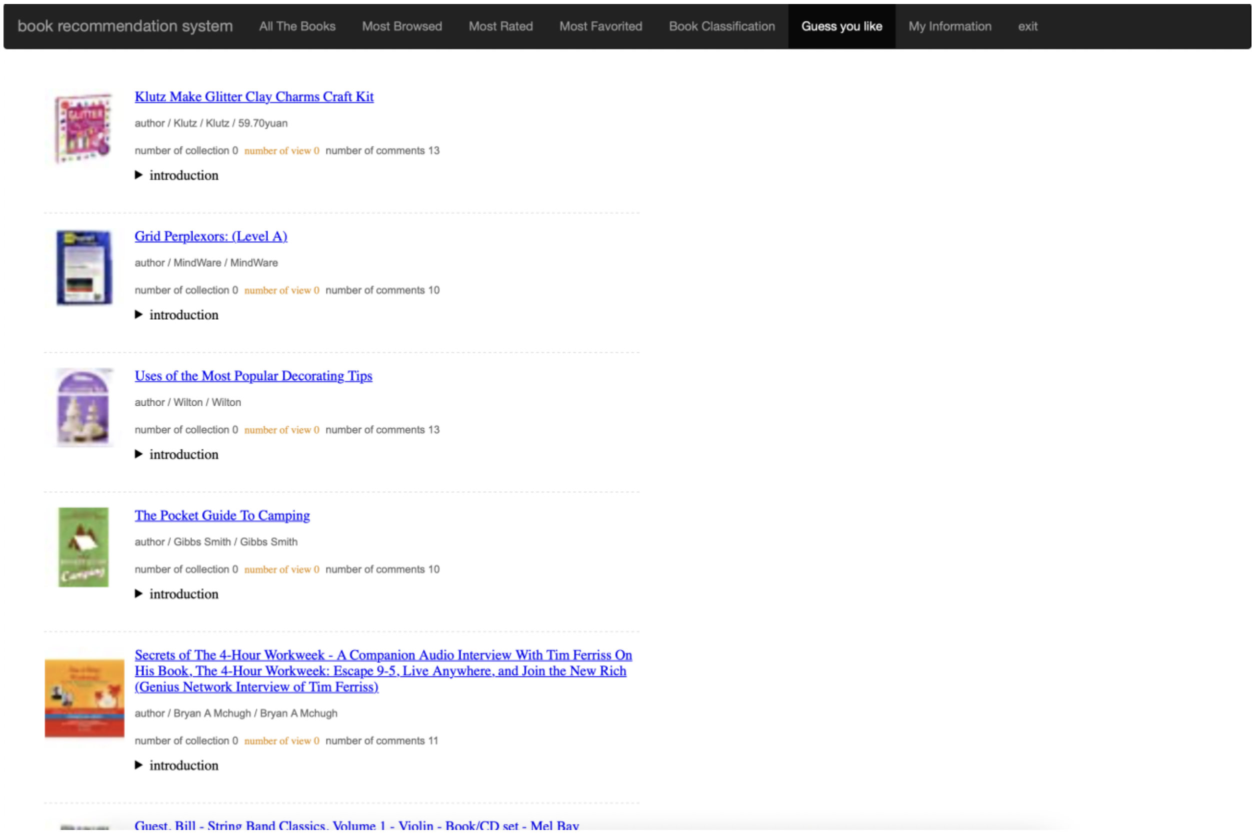

Figure 23: guess you like function

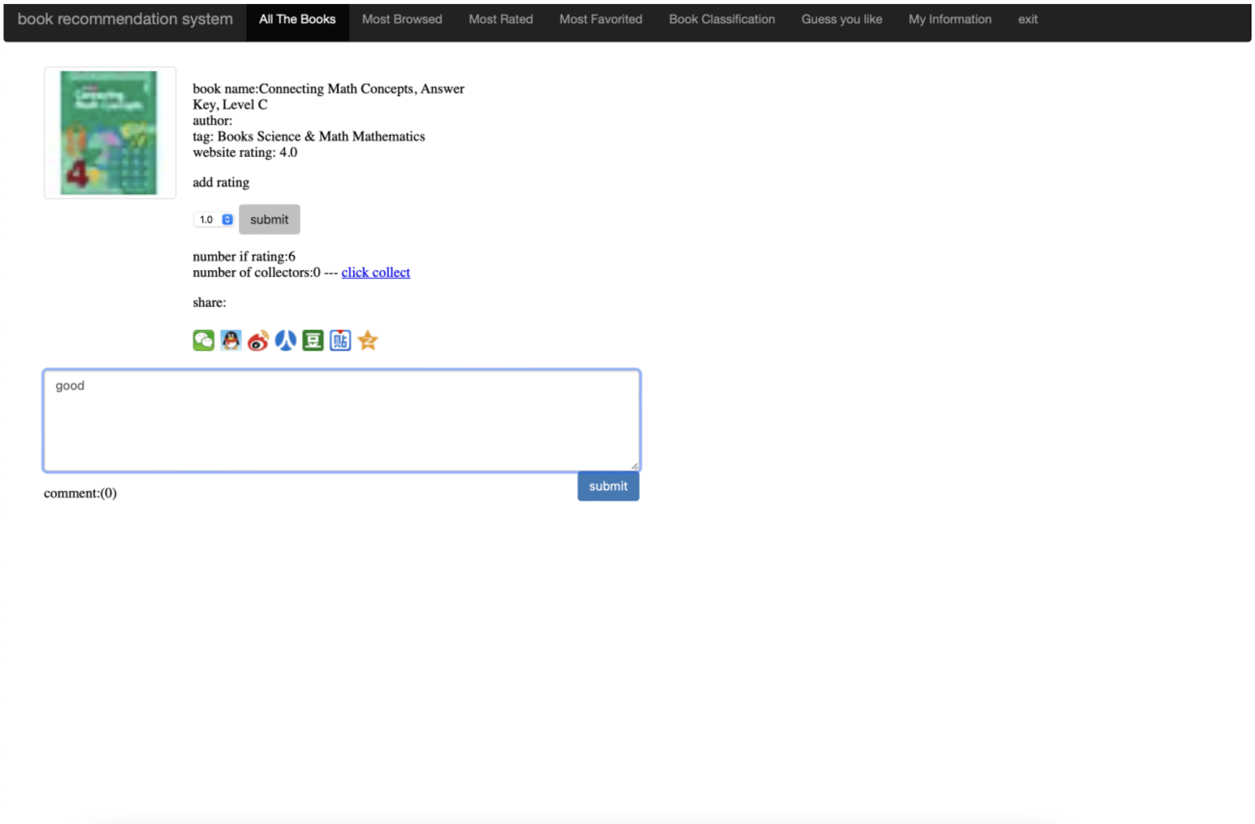

Figure 24: comment function

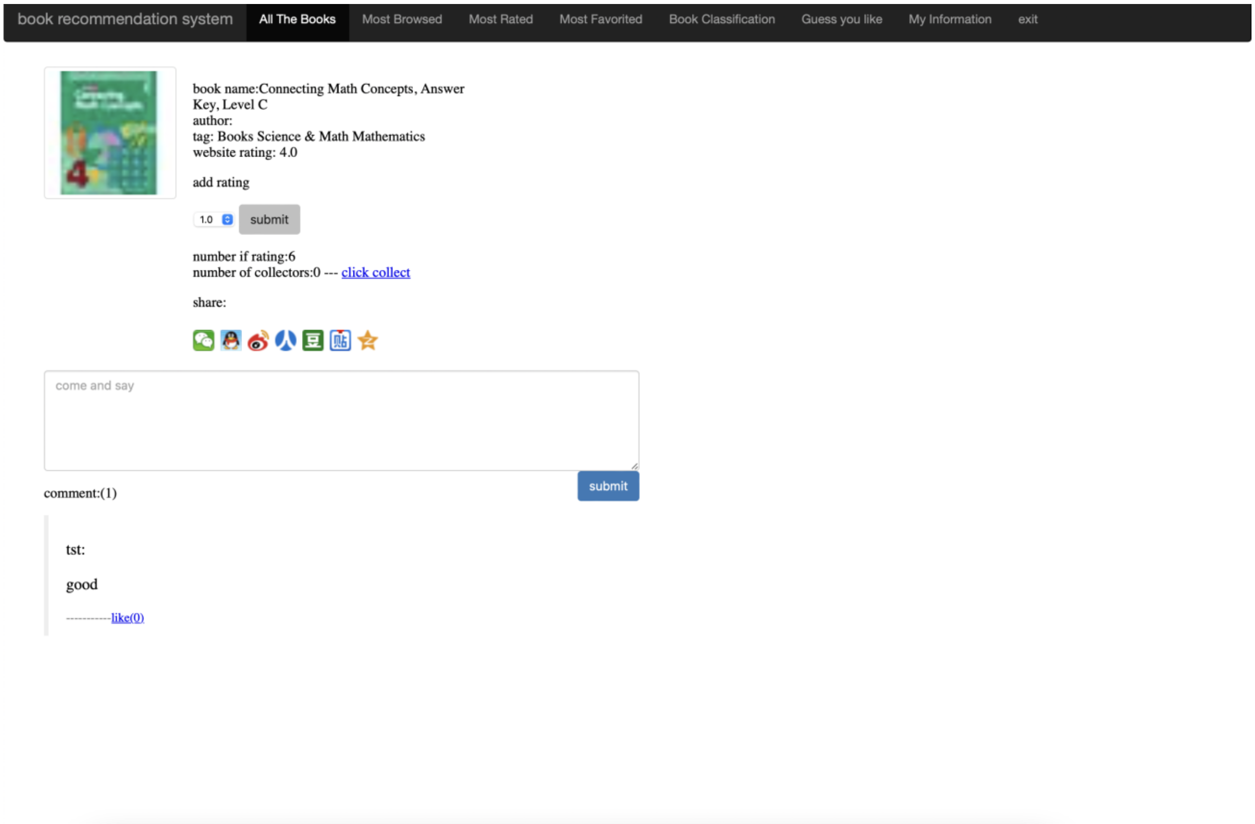

Figure 25: successfully comment

Chapter 5 Professional Issues

5.1 Project Management

5.1.1 Activities

Here are the activities to accomplish each goal:

Object	Activities	complete	uncompleted
A) Research and review, study, and research of the book recommender system.	a) Conduct a systematic search of similar software b) Create a feature comparison table c) Complete a literature search d) Perform a literature review	ALL	NULL

	e) Conduct a user survey		
B) Completes the learning and research of deep learning and explores the book recommendation system.	a) Learning online course of deep learning b) Research of the model c) Collection of the model implements. d) learn the main theory and accomplish a simple one by myself	ALL	NULL
C) Collects appropriate data for analysis and evaluation.	a) Find some articles about processing the data. b) Find the book dataset. c) Analyze the data set, according to different characteristics. d) Divide the dataset into training and testing sets	ALL	NULL
D) Finish data processing with property methods.	a) Check the data format b) Finish the data processing	ALL	NULL
E) Design and implementation of the model and web, using python, Django, HTML, CSS, and JavaScript.	a) Create the Deep & Cross Network (DCN) model. b) Use the training set for model training.	ALL	NULL

	c) Design and create the web base on the Django frame d) Deploy the model with the web application		
F) Test and evaluation of the model and the web application**.**	a) Test model performance by modifying parameters. b) Test web application by function testing and unit testing c) Use the different dataset to test the whole system.	ALL	NULL

Table 2: Activities

5.1.2 **Schedule**

That is my time management and Gantt chart to show the activities and their deadlines:

Task	Start Date	End Date	Duration
Prepare Gantt	2022/10/30	2022/11/7	8
Registration Form and Ethics Form	2022/10/24	2022/10/31	7
Project Proposal	2022/10/26	2022/11/11	16
Annotated bibliography	2022/10/30	2022/12/30	61
Data Collection	2022/11/30	2022/12/24	24
Project Management	2022/10/30	2023/5/5	187
Database Management System Research	2022/11/30	2023/1/1	32
Programming Languages Research	2022/12/1	2023/2/1	62
Interface Design Research	2023/3/16	2023/4/10	25
Progress Report	2022/11/12	2023/1/17	66
Literature Review	2022/10/18	2022/11/3	16
Requirements Analysis	2022/10/31	2022/12/13	43
Model Requirements gathering	2022/11/15	2023/2/10	87
Risk Anlysis	2022/10/18	2022/12/7	50
Code Testing	2022/11/12	2023/1/11	60
Product Testing	2023/1/1	2023/3/20	78
Final Report	2023/1/25	2023/3/20	54
Final Poster	2023/4/1	2023/4/30	29
Presentation demo video	2023/5/1	2023/5/10	9
Design and build	2022/11/15	2023/4/3	139
Implementation	2023/1/1	2023/4/10	99

Figure 26: time management

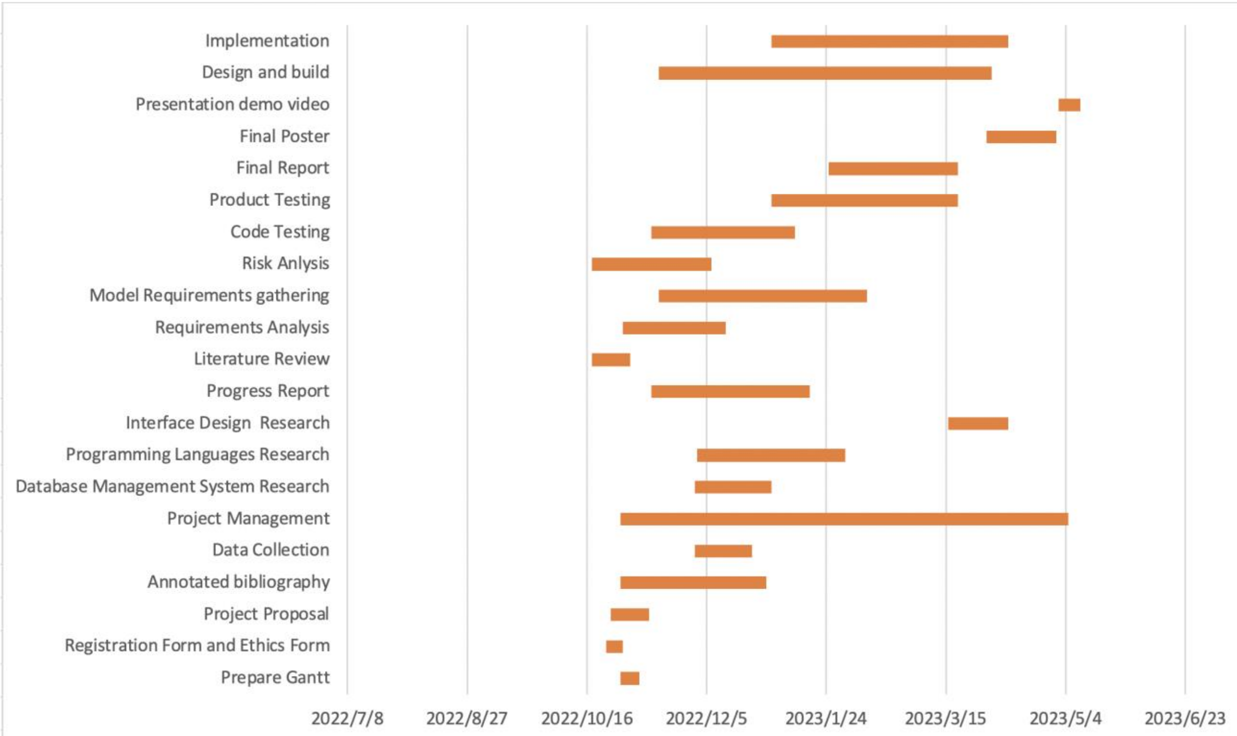

Figure 27: Gantt chart

5.1.3 **Project Data Management**

Create a local folder and upload it through Google Cloud Service

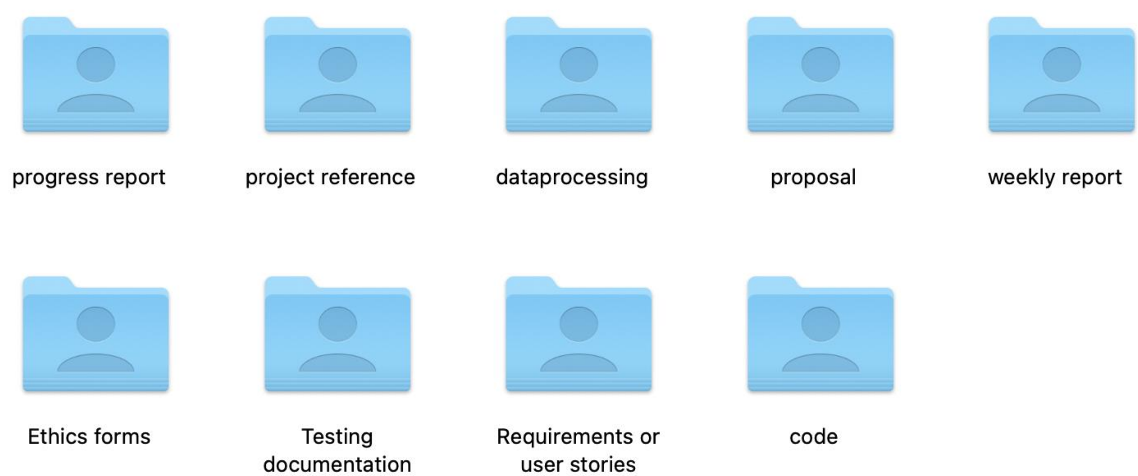

Figure 28: local folder

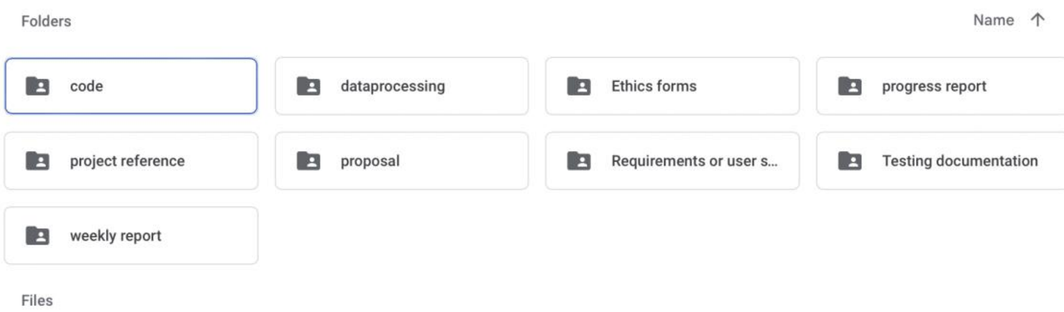

Figure 29: Cloud disk folder

Also, use GitHub to manage the project data:

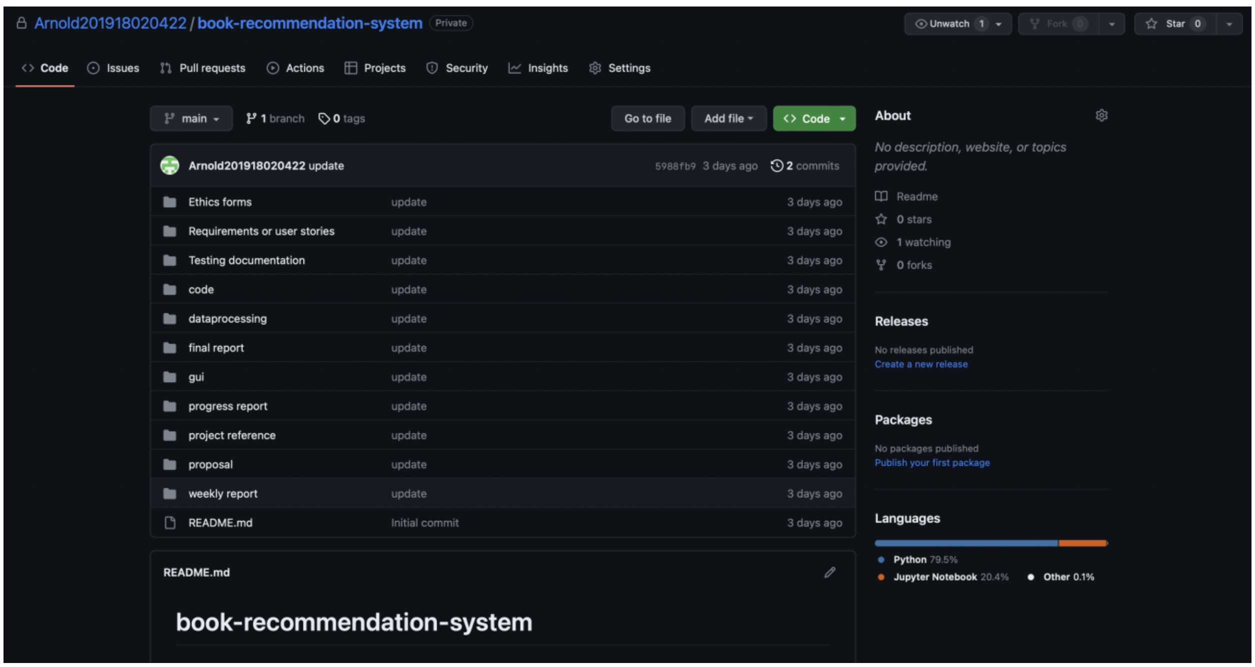

Figure 30: GitHub repository

5.1.4 **Project Deliverables**

Submitted	Not submitted
1. progress report	1. project code
2. project reference	2.testing data

3. dataprocessing	
4. proposal	
5. weekly report	
6. ethics forms	
7. Progress report	
8. testing documentation	
9. final report	
10. dataset	

Table 3: Deliverables

5.2 **Risk Analysis**

The following table presents a list of potential risks to the project, along with their respective likelihood of occurrence, impact on the project, and overall risk score. Additionally, it includes mitigation measures aimed at reducing the impact of the risk or preventing it from happening altogether:

Potential Risk	likelihood	Severity	Risk Score	Potential Causes	Mitigation Strategies
1. Data Quality	High (3)	3	9	Incomplete or inaccurate data	Conduct thorough data cleaning and validation procedures prior to training the model. Select data sources based on their reliability and relevance to the problem domain.
2. Model Complexity	Medium (2)	1	2	High complexity of the Deep and Cross model	Employ thorough testing and validation procedures at each stage of development.

					Allocate resources to ensure adequate computational power and expertise are available for implementation.
3. Overfitting	Medium (2)	3	6	Model becomes too closely fitted to the training data	Use regularization techniques such as dropout and L1/L2 regularization. Evaluate model performance on a holdout validation set to ensure that the model generalizes well to new data.
4. Ethical Concerns	Medium (2)	3	6	Violation of user privacy or perpetuation of harmful stereotypes	Obtain user consent prior to collecting data. Anonymize and encrypt data to protect user privacy. Evaluate fairness and bias in the model design and implementation.
5. Lack of Adequate Data	High (3)	1	3	Insufficient or poor-quality data	Conduct a thorough review of data sources to ensure they are relevant and reliable. Consider using data augmentation techniques to increase the amount of available data.

6. Model Interpretability	Low (1)	2	2	Deep and Cross models can be difficult to interpret	Consider using interpretable models or techniques such as SHAP values to better understand how the model is making recommendations.
7. Algorithmic Bias	Medium (2)	2	4	Bias in the training data or model design	Evaluate the model for bias and take steps to mitigate it. Use diverse and representative data to train the model.
8. Changing User Preferences	High (3)	1	3	User preferences and behaviors can change over time	Consider using techniques such as matrix factorization to update the model with new data. Regularly retrain the model to incorporate changes in user preferences.
9. Scalability	Medium (2)	1	2	Inability of the model to scale effectively	Consider using distributed computing or cloud-based solutions to increase scalability. Regularly monitor the performance of the model as the number of users and items increases.

10.Hyperparameter Tuning	High (3)	2	6	Improper tuning of model hyperparameters	Conduct a thorough hyperparameter tuning process to identify the optimal settings for the model. Consider using techniques such as grid search or random search to identify the optimal values.
11. Data Security	Medium (2)	1	2	Risk of data breaches or unauthorized access	Use appropriate encryption and access controls to protect the data. Regularly monitor the system for security vulnerabilities and take appropriate measures to address them.
12. Legal and Regulatory Compliance	High (3)	2	6	Failure to comply with legal and regulatory requirements	Conduct a thorough review of legal and regulatory requirements and ensure that the system is designed and implemented in compliance with these requirements. Regularly review and update the system to ensure ongoing compliance.

Table 4: risks analysis

By considering these potential risks and implementing the corresponding mitigation strategies, a general ranking based on the likelihood and potential impact of each risk as follows:

1. Data Quality: Poor quality data can severely impact the performance of the recommendation system and is therefore a high-priority risk.
2. Ethical Concerns: Violations of user privacy or the perpetuation of harmful stereotypes can have significant legal and reputational consequences, making this a high-priority risk.
3. Algorithmic Bias: Bias in the model or training data can impact the accuracy and fairness of the recommendations and is therefore a high-priority risk.
4. Model Complexity: The complexity of the Deep and Cross model can lead to issues such as overfitting and longer development times, making this a moderate-priority risk.
5. Lack of Adequate Data: Insufficient or poor-quality data can impact the performance of the recommendation system and is therefore a moderate-priority risk.
6. Model Interpretability: Lack of interpretability can hinder the ability to explain the recommendations but may not be as critical as other risks depending on the stakeholders involved.
7. Changing User Preferences: Changes in user preferences can impact the accuracy of the recommendations but can be addressed through regular retraining of the model.
8. Scalability: Inability to scale may be a concern for larger systems or user bases but can be addressed with distributed computing or cloud-based solutions.
9. Hyperparameter Tuning: Improper tuning of hyperparameters can lead to suboptimal model performance but can be addressed through thorough testing and validation procedures.
10. Data Security: While important, data security risks can be addressed through appropriate encryption and access controls, making it a lower-priority risk.
11. Legal and Regulatory Compliance: Compliance with legal and regulatory requirements is important but can be addressed through a thorough review and ongoing monitoring and updates, making it a lower-priority risk.

After conducting a risk analysis of the project, this article has identified multiple potential risk factors. Firstly, there is a data quality issue, followed by model complexity and overfitting issues. Ethical issues also need to be taken seriously, such as sensitive personal information and discriminatory algorithms. A lack of sufficient data may also make it difficult for the model to learn and predict accurately. Model interpretability and algorithmic bias should also be considered, as well as demand risk if the system cannot meet user needs or adapt to changing user needs. Finally, insufficient scalability of the system should also be considered.

To mitigate these potential risks, the following measures are recommended: (1) using effective data cleaning and preprocessing techniques to eliminate interference factors in the data; (2) adopting appropriate regularization methods, adding more training data, etc. to address model complexity and overfitting issues; (3) paying close attention to ethical issues and taking corresponding measures to ensure algorithm fairness and transparency; (4) evaluating whether there is enough data to support training and testing, and using transfer learning, etc. to solve data scarcity problems; (5) using interpretable models, increasing the diversity of training data, etc. to address model interpretability and algorithmic bias issues; (6) communicating and researching with users to understand their needs and expectations and adjusting the system design in a timely manner to reduce demand risk; (7) considering scalability in system design and taking measures to ensure that the system can support future business needs.

Therefore, through careful risk analysis and effective risk management measures, the impact of these risks can be minimized, and the development of efficient, reliable, and secure machine learning models can be ensured, bringing more value to our lives and work. At the same time, data security, compliance, and other issues should also be taken seriously during development, complying with relevant laws and ethical standards to ensure the sustainability and long-term development of the project.

5.3 **Professional Issues**

Professional Issue	**Potential Causes**	**Mitigation Strategies**	**Relevant Laws and Regulations**
Legal Issues	Use of Amazon dataset without permission or infringing on intellectual property rights	Obtain necessary permissions and licenses, and adhere to all relevant laws and regulations, such as data protection laws and intellectual property rights	General Data Protection Regulation (GDPR), Copyright Law

Social Issues	Use of user data without proper consent or transparency	Ensure that users are informed of how their data will be used, and obtain their consent before using it. Additionally, consider the potential impact of the system on different user groups, and strive to avoid any unfair treatment or biases.	GDPR, e-Privacy Regulation, Fair Credit Reporting Act (FCRA)
Ethical Issues	Bias in the recommendation system due to the use of historical user data or other factors	Design the recommendation system to avoid any biases or unfair treatment of certain users or groups. Additionally, ensure that the system is transparent and understandable to users, and that users are made aware of how the recommendations are generated.	ACM Code of Ethics and Professional Conduct, IEEE Code of Ethics
Environmental Issues	Energy consumption and environmental impact of training and running the deep learning model	Consider the energy consumption and environmental impact of training and running the deep learning model, and implement strategies to reduce the impact, such as using more energy-efficient hardware or optimizing the training process.	Energy Star, EPEAT

Table 5: professional issues

1. Use of Amazon dataset without permission or infringing on intellectual property rights:
In a study on e-commerce recommendation systems, researchers wanted to evaluate the performance of their system using a large-scale product catalog. They decided to use the Amazon dataset without obtaining proper consent or licensing from Amazon. However, this violated Amazon's intellectual property rights, and the researchers could face legal consequences for their actions.

2. Use of user data without proper consent or transparency:
A group of researchers conducting a study on social media behavior collected user data through a browser extension without informing users about the extent of data being collected. This violates ethical standards of informed consent and transparency, and it could result in negative publicity and even legal action against the researchers.

3. Bias in the recommendation system due to the use of historical user data or other factors:
In a study evaluating the effectiveness of an algorithmic hiring tool, researchers trained the model on historical employee data. However, these data reflected biases in the hiring process such as gender and race-based discrimination. As a result, the algorithm recommended fewer candidates from minority groups, which could lead to discriminatory practices in the workplace.

4. Energy consumption and environmental impact of training and running the deep learning model:
Researchers developing deep learning models may need to train their models using high-end hardware and cloud computing services, which consume significant amounts of energy. For example, a study on natural language processing using transformer models found that training a single model could consume hundreds of kilowatt-hours of electricity. The environmental impact of such large-scale energy consumption can be significant, and researchers need to consider more energy-efficient alternatives.

Chapter 6 Conclusion

This study presents a book recommendation system based on the deep learning model DCN, which is trained on a substantial amount of book data to handle high-dimensional sparse feature data and improve recommendation accuracy. The system is showcased through a web application that offers basic registration functionality and implements the DCN model's online service to update recommendation results by analyzing user history and interests. Our experimental results demonstrate that the DCN-based recommendation system outperforms traditional collaborative filtering algorithms in terms of accuracy and efficiency.

I propose further improvements in feature engineering and model optimization for future work to enhance recommendation accuracy and speed. Additionally, I want to explore the application of other deep learning models in the book recommendation field to achieve more precise and accurate recommendation services. I think that collecting user feedback and behavioral data to optimize their experience is important in continuously improving the system.

Moreover, I realized the incorporation of diverse and personalized features into the recommendation system is acceptable, such as demographic information, reading preferences, and social network data, to tailor recommendations to each individual user's unique tastes and preferences. Furthermore, integrating the book recommendation system with other platforms or services, such as e-commerce websites, social media platforms, or mobile applications, could provide users with a seamless and integrated experience, facilitating the discovery and purchase of books based on their interests and previous reading history.

In conclusion, the development of this DCN-based book recommendation system, along with its accompanying web application, represents a significant advancement in the field of personalized recommendation systems. We believe that continued research and development will lead to even more powerful and accurate models, providing users worldwide with an enhanced reading experience.

References

[1] L. E. Pilli and J. A. Mazzon, "Information overload, choice deferral, and moderating role of need for cognition: Empirical evidence," *Revista de Administração*, vol. 51, no. 1, pp. 036–055, 2016, doi: 10.5700/rausp1222.

[2] P. Resnick and H. R. Varian, "Recommender Systems," 1997. [Online]. Available: http://www.firefly.com

[3] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Comput*, vol. 7, no. 1, pp. 76–80, 2003, doi: 10.1109/MIC.2003.1167344.

[4] G. Linden, B. Smith, and J. York, "Amazon.com Recommendations," IEEE Computer Society. [Online]. Available: http://computer.org/internet/

[5] "Review of Movie Recommendation System | Enhanced Reader 2 0 8t h."

[6] A. Singhal, P. Sinha, and R. Pant, "Use of Deep Learning in Modern Recommendation System: A Summary of Recent Works," 2017.

[7] R. Wang, G. Fu, B. Fu, and M. Wang, "Deep & cross network for ad click predictions," in *2017 AdKDD and TargetAd - In conjunction with the 23rd ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2017*, Association for Computing Machinery, Inc, Aug. 2017. doi: 10.1145/3124749.3124754.

[8] "PRINCIPLES OF FORECASTING: A Handbook for Researchers and Practitioners."

[9] M. Sun, G. Lebanon, and P. Kidwell, "Estimating Probabilities in Recommendation Systems," 2011.

[10] H. Liu, J. He, T. Wang, W. Song, and X. Du, "Combining user preferences and user opinions for accurate recommendation," *Electron Commer Res Appl*, vol. 12, no. 1, pp. 14–23, 2013, doi: 10.1016/j.elerap.2012.05.002.

[11] G. Adomavicius and A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions."

[12] "mis.2002.1134360".

[13] K. Verbert *et al.*, "Context-aware recommender systems for learning: A survey and future challenges," *IEEE Transactions on Learning Technologies*, vol. 5, no. 4. pp. 318–335, 2012. doi: 10.1109/TLT.2012.11.

[14] R. J. Mooney, "Content-Based Book Recommending Using Learning for Text Categorization," 2000.

[15] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," 1998.

[16] "ecom Syst." [Online]. Available: http://fab.stanford.edu.

[17] H. T. Cheng *et al.*, "Wide & deep learning for recommender systems," in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Sep. 2016, pp. 7–10. doi: 10.1145/2988450.2988454.

[18] L. Zhang, T. Luo, F. Zhang, and Y. Wu, “A Recommendation Model Based on Deep Neural Network,” *IEEE Access*, vol. 6, pp. 9454–9463, Jan. 2018, doi: 10.1109/ACCESS.2018.2789866.

[19] X. Xu, “Matrix factorization recommendation algorithm based on deep neural network,” in *2019 2nd International Conference on Information Systems and Computer Aided Education, ICISCAE 2019*, Institute of Electrical and Electronics Engineers Inc., Sep. 2019, pp. 320–323. doi: 10.1109/ICISCAE48440.2019.221643.

[20] P. Covington, J. Adams, and E. Sargin, “Deep neural networks for youtube recommendations,” in *RecSys 2016 - Proceedings of the 10th ACM Conference on Recommender Systems*, Association for Computing Machinery, Inc, Sep. 2016, pp. 191–198. doi: 10.1145/2959100.2959190.

[21] Y. Zhang, M. Pezeshki, P. Brakel, S. Zhang, C. L. Y. Bengio, and A. Courville, “Towards End-to-End Speech Recognition with Deep Convolutional Neural Networks,” Jan. 2017, [Online]. Available: http://arxiv.org/abs/1701.02720

[22] S. Okura, Y. Tagami, S. Ono, and A. Tajima, “Embedding-based news recommendation for millions of users,” in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Association for Computing Machinery, Aug. 2017, pp. 1933–1942. doi: 10.1145/3097983.3098108.

Appendices