Online music personalized recommendation algorithm based on KKBox large-scale real-world dataset and integrated behavior sequence transformer

Mingchuan Zhong, Xinyi Lan

[1]*School of Information Management and Mathematics, Jiangxi University of Finance and Economics, Nanchang, Jiangxi, China*

[2]*School of Computer and Artificial Intelligence, Jiangxi University of Finance and Economics, Nanchang, Jiangxi, China*

Received: 15 Aug 2025
Revised: 19 Aug 2025
Accepted: 27 Aug 2025
Published: 29 Aug 2025

Abstract: To address the user selection overload and decision-making difficulties caused by the explosive growth of online music platforms, this paper proposes and validates a deep learning recommendation model that incorporates the Behavior Sequence Transformer (BST), aiming to significantly improve music recommendation accuracy, diversity, and novelty. We integrate 7.3 million user-song interaction records from the KKBox 2004–2018 public and crawled dataset to construct a 16-dimensional feature system encompassing user, song, and context. We use the BST-Behavior-Sequence-Transformer to perform multi-head self-attention modeling on user listening sequences and compare it with a LightGBM boosted tree and logistic regression baseline model. Experiments use AUC, accuracy, and RMSE as core metrics, and use an 80/20 train-test split and early stopping to control overfitting. The BST model achieved an AUC of 0.7662 and an accuracy of 75.51% on the test set, significantly outperforming LightGBM (AUC of 0.6579 and an accuracy of 64.69%) and logistic regression (AUC of 0.6871 and an accuracy of 67.68%). Feature importance analysis shows that user-genre cross-features, contextual entry points, and sequence position encoding contribute most to the model. The recommendation algorithm based on the Behavior Sequence Transformer effectively captures the dynamic evolution of user interests, alleviates information silos, and possesses good scalability and engineering potential. Future research into multimodal fusion and reinforcement learning can further enhance diversity and interpretability.

Keywords: Music recommendation system; Behavior Sequence Transformer; Attention Mechanism; LightGBM; KKBox dataset; Deep Learning

1 INTRODUCTION

In recent decades, with the rapid development of information technology and the widespread adoption of the internet, music consumption has undergone profound changes. The traditional music distribution model, once dominated by radio DJs, has gradually declined,

replaced by digital music listening centered around online streaming platforms. These platforms, with their vast music resources and high accessibility, enable users to access personalized audio content anytime, anywhere. Existing research shows that mainstream music service platforms (Such as KKBOX, NetEase Cloud Music, and QQ Music) typically have music libraries exceeding 40 million songs. Taking QQ Music as an example, if each song averages three minutes, it would take over 170 years of continuous listening to complete its catalog. Faced with such a vast amount of music, users often experience information overload and decision-making difficulties when selecting music, making it difficult to efficiently discover tracks that suit their preferences.

To address this issue, music recommendation systems have emerged. These systems rely on algorithms to analyze music content characteristics and user history to proactively provide users with tracks that may be of interest. As early as 1995, the Ringo system, the first music recommendation algorithm, was proposed. It could predict users' ratings for specific songs, marking the beginning of automated music recommendations. Early recommendations largely relied on track metadata and lacked true personalization. Subsequently, platforms such as Pandora and Last.fm advanced music recommendation. For example, Pandora, leveraging its "Music Genome Project," quantified song features and calculated similarities to enable content-based recommendations.

Currently, mainstream online music platforms widely integrate music recommendation algorithms to enhance user experience. For example, QQ Music, through features like "Radar Playlist" and "Daily Recommendations," leverages users' listening history, social behavior, and song metadata to identify groups of users with similar preferences and generate recommendation lists. Despite this, most existing systems still rely on collaborative filtering methods, which analyze users' historical behavior (Such as play and favorite records) and their similarity to other users to make music recommendations. With the increasing diversity of user aesthetics and the continuous expansion of music listening scenarios, these traditional methods are gradually becoming limited in terms of accuracy, diversity, and novelty. Therefore, in the context of extremely abundant music resources and rising user expectations, building more accurate, efficient, and adaptive music recommendation algorithms has become a key issue in improving platform service quality and user satisfaction. This study aims to systematically explore the optimization paths for music recommendation strategies and provide theoretical support and practical references for their further development.

2 RECOMMENDATION SYSTEM AND ALGORITHM PRINCIPLES

2.1 Recommender System Architecture

A recommendation system can proactively recommend items or services that a user might be interested in, even without explicit user input, by analyzing historical user behavior, social connections, context, item attributes, and review text. The core of a recommendation system is the recommendation algorithm, and choosing the right method to build a recommendation model is a crucial aspect of a recommendation system.

2.2 Traditional Recommendation Algorithms

Traditional recommendation algorithms include content-based recommendation patterns, rule-based recommendation, and collaborative filtering. Content-based filtering relies on item metadata to calculate item similarity and then recommends similar items to the user based on historical user behavior. Rule-based recommendation is commonly used in e-commerce systems, extracting association rules from large amounts of transaction data or sequential models of item purchases over time to recommend items to each other. Collaborative filtering includes user-based and item-based collaborative filtering. User-based collaborative filtering analyzes user historical behavior to calculate similarity between users and then uses this similarity and historical behavior to create a recommendation list for the user. Item-based collaborative filtering is similar to this approach. It calculates the similarity between items by analyzing user behavior and then recommends similar items to users based on their historical preferences.

2.3 Recommendation Algorithm Based on the BST-Behavior-Sequense-Transformer Model

Considering that traditional recommendation algorithms may no longer perform well, we have collected extensive research and referenced the latest attention mechanism + deep learning methods. We are planning to experiment with the BST-Behavior-Sequense-Transformer algorithm model as a recommendation model. The algorithm framework is as follows:

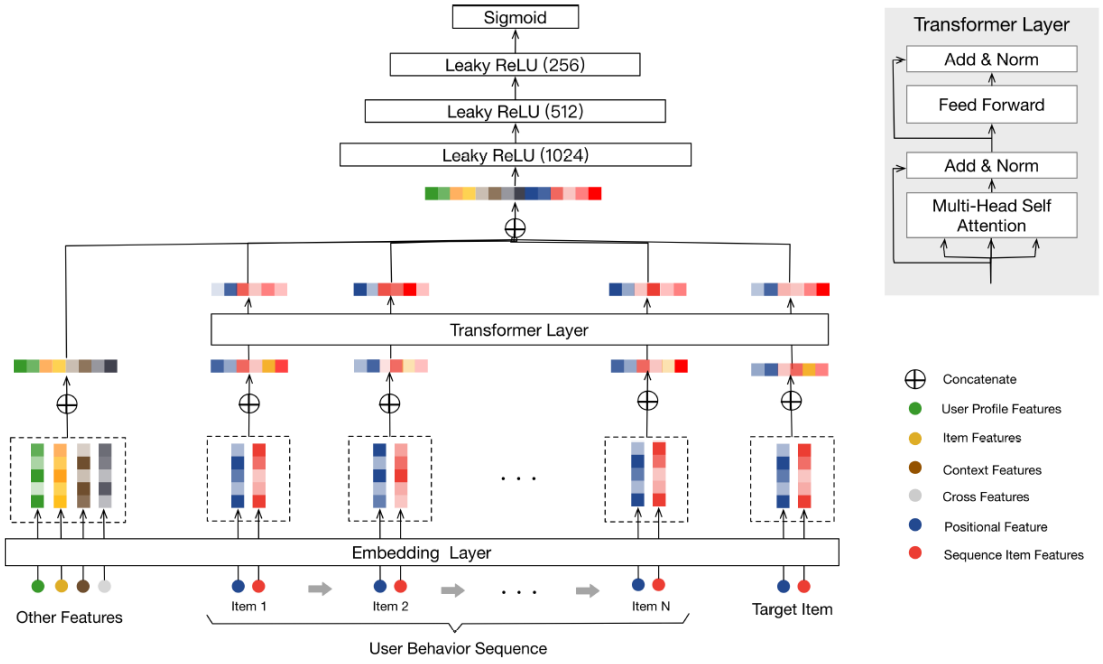

Fig. 1: BST-Behavior-Sequense-Transformer algorithm model

The BST (Behavior-Sequense-Transformer) algorithm, inspired by the tremendous success of the Transformer in machine translation tasks in natural language processing, incorporates a multi-head self-attention mechanism to learn a better representation for each item in a user's behavior sequence, while also taking into account sequential information in the embedding stage [1].

Therefore, we chose the User Behavior Sequence Transformer (BST) approach for our online music recommendation research.

The overall architecture is as follows: In the ranking stage, we model the recommendation task as a click-through rate (CTR) prediction problem, where the input consists of a user's music

listening behavior sequence and other features. These input features are first embedded into low-dimensional vectors. To better capture the relationships between songs in the behavior sequence, Transformer layers are used to learn a deeper representation of each item in the sequence [2]. Then, by concatenating the embeddings of other features and applying the Transformer layer to the target song, three fully-connected layers are used to further learn the interactions between dense features.

The three core layers of the algorithm are briefly described below:

Embedding Layer:

The first component is the Embedding layer, which embeds all input features into a fixed-size low-dimensional vector. Transformer Layer

The Transformer layer uses a self-attention mechanism to capture the relationships between items in the behavior sequence. Furthermore, following multi-head attention, we add a point-wise feedforward network (FFN) to further enhance the model's nonlinearity. To avoid overfitting and enable layer-by-layer learning of meaningful features, we employ dropout and LeakyReLU in both the self-attention and FFN layers [3]. The overall outputs of the self-attention and FFN layers are then combined using layer-by-layer normalization.

MLP Layer：

By concatenating the embeddings of other features with the output of the Transformer layer applied to the target item, we use three fully-connected layers to further learn the interactions between dense features. The output of the final sigmoid fully-connected layer is used as the output unit. To train the model, we experimentally use a cross-entropy loss.

2.4 Recommendation Algorithm Based on the LightGBM Decision Tree Model

The evolution of decision tree algorithms has progressed from C3.0 (based on information gain) to CART (based on the Gini index) to AdaBoost to Gradient Boosted Trees (GDBT) to XGBoost, and finally to the current LightGBM algorithm. Among them, LightGBM (Light Gradient Boosting Machine) is a distributed gradient boosting framework based on the decision tree algorithm that was open-sourced by Microsoft in August 2017. Compared with previous boosting frameworks, it has faster training efficiency, lower memory usage, higher accuracy, supports parallel learning, and can handle large-scale data. It can be used for sorting, classification, and many other machine learning tasks [4].

Recommendation algorithms based on decision tree models have the following advantages: (1) they can be trained in parallel; (2) they can handle discrete and continuous feature values and categorical features without normalizing the features; (3) they can handle missing values; and (4) they can handle high-dimensional features.

So below, we also use the LightGBM decision tree model recommendation algorithm to observe its music recommendation effect in the large data set we collected.

3 DATASET SELECTION AND ANALYSIS

3.1 Dataset Selection and Collection

KKBox is Asia's leading music streaming service, founded in 2005, boasting the world's most comprehensive library of Asian pop music. It leverages innovative cloud-based technology to provide music streaming services, allowing users to play songs stored in the cloud over the internet. Furthermore, it employs DRM (Digital Rights Management) technology to encrypt media files, successfully achieving a perfect balance and protection between music and intellectual property rights [5]. This has revolutionized the concept of legal copyright licensing for online music, and has made KKBox a benchmark brand in the Asian

market.

To date, KKBox boasts over 40 million tracks and serves Taiwan, Hong Kong, Japan, Singapore, and Malaysia. By tapping into this vast library of historical music listening records, streaming services like KKBox can provide users with more personalized music recommendations. Therefore, in this experiment, we selected and collected songs and user information from the KKbox music platform from 2004 to 2018, using this data to form the music dataset required for our experiment. (Since the KKbox music platform already provides a publicly available music dataset from 2004 to 2017, we used this dataset directly and then crawled some relevant data from 2017 to 2018 using the provided dataset format.) By analyzing the collected user-music data and applying data mining techniques, we constructed a suitable algorithmic model to accurately recommend music tracks [6].

3.2 Data Description

The music dataset used in this experiment is divided into four parts, stored in the following data files: recommendations.csv, songs.csv, members.csv, and songs_name.csv. These data tables store various fields of collected information (of course, some fields may be repeated, but some fields are grouped together for easier analysis). The following describes the meaning of the fields contained in each data file.

Table 1: Field information stored in the recommendation file

Field	Meaning
msno	User ID (encrypted)
song_id	Song ID
source_system_tab	The name of the tab that triggered the listening event. System tabs are used to categorize KKBOX mobile app features. For example, the "My Library" tab contains functions for accessing local storage, while the "Search" tab contains search-related functions.
source_screen_name	
source_type	Name of the page the user sees
target	

This data table can be thought of as a user log data table. It stores fields related to both users, such as user IDs, and songs, such as song IDs. It primarily stores information describing user actions and how they listened to songs. For subsequent experiments, the entire recommendation.csv dataset was divided into an 80% training set (train) and a 20% test set (test).

The collected user ID fields (msno) are encrypted, as shown in Figure 2 below. The encrypted user fields display a long string of characters.

```
In [22]: members_data.head()
Out[22]:
                                          msno  ...  expiration_date
0  XQxgAYj3klVKjR3oxPPXYYFp4soD4TuBghkhMTD4oTw=  ...       2017-09-20
1  UizsfmJb9mV54qE9hCYyU07Va97c0lCRLEQX3ae+ztM=  ...       2017-06-22
2  D8nEhsIOBSoE6VthTaqDX8U6lqjJ7dLdr72mOyLya2A=  ...       2017-07-12
3  mCuD+tZ1hERA/o5GPqk38e041J8ZsBaLcu7nGoIIvhI=  ...       2015-09-07
4  q4HRBfVSssAFS9iRfxWrohxuk9kCYMKjHOEagUMV6rQ=  ...       2017-06-13
```

Fig. 2: Encrypted user ID field

The following is the field information for the songs.csv table:

Table 2: Field information stored in the songs file

Field	Meaning
song_id	Song ID
song_length	Song duration
genre_ids	Song genre ID. However, some songs may have multiple genres, so we use the | symbol to separate them.
artist_name	Singer
composer	Composer
lyricist	Lyricist
language	Language ID

Table 3: Field information stored in the members file

Field	Meaning
msno	User ID (also encrypted)
city	User's city ID
bd	Age
gender	Gender
registered_via	Registration method
registration_init_time	Registration time (in the format of %Y%m%d)
expiration_date	Expiration time (also in the format of %Y%m%d)

Table 4: Field information stored in the songs_name file

Field	Meaning
songs_id	Song ID
song_name	Song name
isrc	International Standard Recording Code, also a song identifier

Looking at the selected and collected field data, the entire dataset is actually divided into three dimensions: user, music, and operation.

3.3 Data Analysis

The user information table contains 34,403 sample data entries, as shown in Figure 3 below:

```
In [23]: members_data.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34403 entries, 0 to 34402
Data columns (total 7 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   msno                   34403 non-null  object
 1   city                   34403 non-null  int64
 2   bd                     34403 non-null  int64
 3   gender                 14501 non-null  object
 4   registered_via         34403 non-null  int64
 5   registration_init_time 34403 non-null  datetime64[ns]
 6   expiration_date        34403 non-null  datetime64[ns]
dtypes: datetime64[ns](2), int64(3), object(2)
memory usage: 1.8+ MB
```

Fig. 3: User Information Table Data

It can be seen that except for the gender field, all other fields have no missing values. However, the gender field only has 14,501 entries, resulting in a missing rate of 57.85%, exceeding half of the total. Furthermore, the gender entered by users during registration may not always be correct [7]. Further examining the other fields, we find that while the age field has no missing values, its distribution reveals that the majority of the data is zero, with a maximum of 1051 and a minimum of -43.

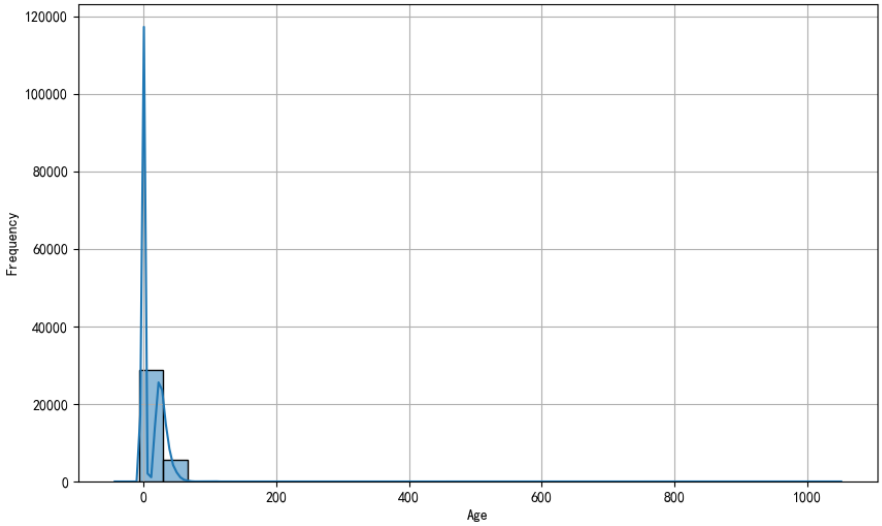

Fig. 4: User Age Distribution

Examining age information, we see 10,377 users aged 0, which can be considered missing data. Furthermore, the majority of age distribution falls between 18 and 30, showing a roughly normal distribution with minimal differentiation.

Further examining the city field reveals the distribution of users by city.

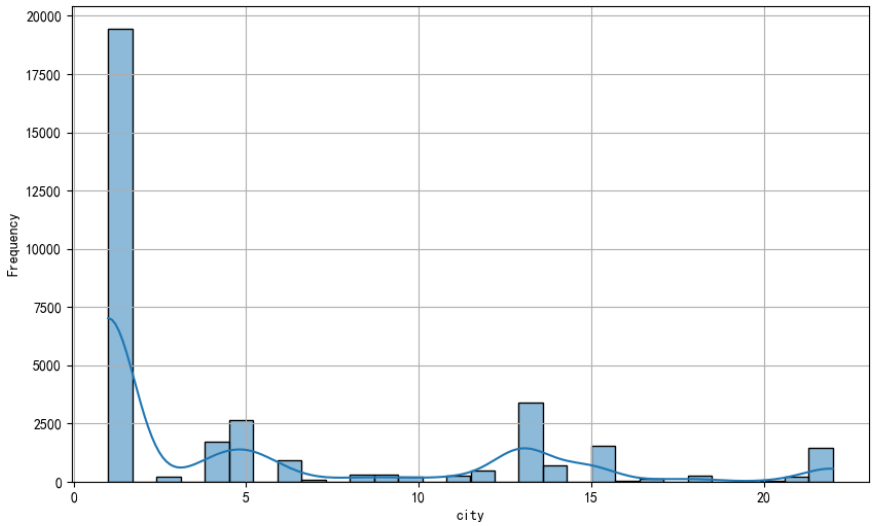

Fig. 5: User City Distribution

From the above distribution, we can see that the majority of users are located in city 1. There are 19,445 records with city 1, which is somewhat unrealistic. Considering the previous users with age 0 and gender blank, we believe these users skipped the registration process

without entering any information [8]. Therefore, city 1, age 0, and gender are missing. Therefore, these three values can be treated as missing.

Regarding the registration time format, which is %Y%M%D, we convert it to a continuous variable representing the number of days registered. We also count the number of plays and repetition rate for each feature element and add it to the user information table.

Similarly, we perform similar data processing and analysis on the music information table, first observing the statistical description of the song metadata, as shown in Figure 5 below.

```
In [5]: songs_data.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2296320 entries, 0 to 2296319
Data columns (total 7 columns):
 #   Column       Dtype
---  ------       -----
 0   song_id      object
 1   song_length  int64
 2   genre_ids    object
 3   artist_name  object
 4   composer     object
 5   lyricist     object
 6   language     float64
dtypes: float64(1), int64(1), object(5)
memory usage: 122.6+ MB
```

Fig. 6: Statistical Description of Song Metadata

There are a total of 2,296,320 data samples. Similarly, analysis of each field revealed that the genre_ids field is missing 485 entries, a missingness rate of 1.6%. This means that the genres of all 485 songs are unknown. Frequency distribution reveals that the most frequent entry is 465 (the song genre ID), which occurs 16,735 times, accounting for 50% of the songs. Therefore, for samples with missing values in the genre_ids field, the most frequent entry can be used to fill in the missing values. Further analysis of this field reveals that approximately 80% of songs have only one genre ID, while the rest have at most two categories [9]. A few songs have multiple genre IDs. Given the importance of song genre characteristics, it may be appropriate to use CountVectorizer to convert the data into a sparse matrix and not handle missing values. Observing other fields, we found that composer has 1,071,354 missing entries, a missingness rate of approximately 46.66%. Finally, lyricist has 1,945,268 missing entries, a missingness rate of approximately 84.71%. Overall, the missing data for lyricist and composer data is too high, so we might consider deleting these two fields. Interestingly, there's a missing value for language [10]. After searching, we discovered it was a data misplacement, so we manually filled in the value.

As for the song_length and language fields, our experiments revealed that most songs are between 2 and 4 minutes long, and the language is mostly number 52. After discussion, we believe this likely represents Mandarin.

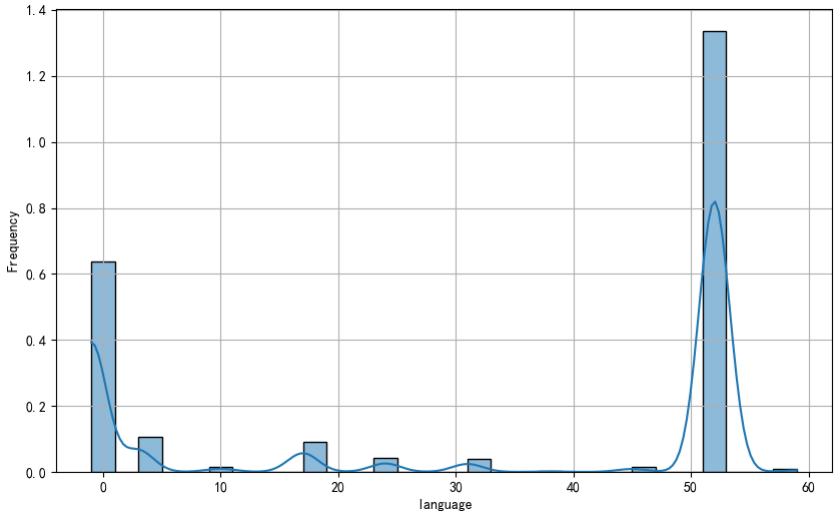

Fig. 7: Distribution of Song Language Field Values

4.3.3 Song Additional Information Table

Descriptive statistics were also performed on this dataset, with the results shown in Figure 8 below.

```
In [8]: song_extra_info_data.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2295971 entries, 0 to 2295970
Data columns (total 3 columns):
 #   Column   Dtype
---  ------   -----
 0   song_id  object
 1   name     object
 2   isrc     object
dtypes: object(3)
memory usage: 52.6+ MB

In [9]: song_extra_info_data.describe()
Out[9]:
                                          song_id     name          isrc
count                                     2295971  2295968       2159423
unique                                    2295971  1168978       1806825
top     LP7pLJoJFBvyuUwvu+oLzjT+bI+UeBPURCecJsX1jjs=   Intro  GBPS81518952
freq                                            1     1734           207
```

Fig. 8: Descriptive Statistics of the Song Extra Information Dataset

In the song extra information (song_extra_info.csv), the name column has 3 missing data entries, with a missing rate of approximately 0.000087%, and the isrc column has 136,548 missing data entries, with a missing rate of approximately 5.95%. However, when predicting user preferences, song names are not necessary, as song IDs uniquely identify songs, so this column can be deleted [11]. The song's audiovisual product code can be converted into the song's release date for use in time series models.

Since the listening history table has already been partitioned for the experiment, the following descriptive statistics are performed on the train dataset, using it as an example.

```
In [10]: train_data.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7377418 entries, 0 to 7377417
Data columns (total 6 columns):
 #   Column              Dtype
---  ------              -----
 0   msno                object
 1   song_id             object
 2   source_system_tab   object
 3   source_screen_name  object
 4   source_type         object
 5   target              int64
dtypes: int64(1), object(5)
memory usage: 337.7+ MB
```

Fig. 9: Descriptive statistics of the train dataset.

In the user listening records (train.csv), 24,849 entries in the source_system_tab column are missing, representing a missing rate of approximately 0.34%. 414,804 entries in the source_screen_name column are missing, representing a missing rate of approximately 5.62%. Finally, 21,539 entries in the source_type column are missing, representing a missing rate of approximately 0.29%. Overall, the missing data ratio is not high, and data completion can be achieved through data imputation.

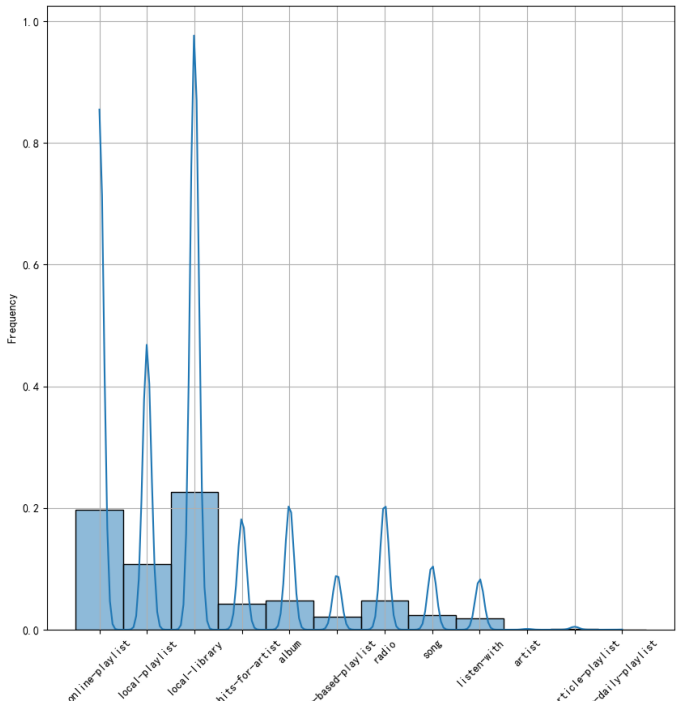

Fig. 10: Distribution of source_type field values.

In addition, observing the distribution of other fields reveals that the target label is evenly distributed, indicating that the number of samples in different classes is very balanced, and no special processing is required for minority class samples. The test.csv data is processed in the same way as the train.csv dataset, which will not be discussed in detail here.

4 FEATURE ENGINEERING

After completing dataset selection and preliminary analysis, systematic data processing is required to extract effective information from the raw, complex data to support modeling. This section details methods for handling missing values and outliers, aiming to improve data quality and feature reliability.

Preliminary analysis revealed numerous missing fields in the user and song information tables, particularly in the gender, composer, and lyricist fields. Considering that composer and lyricist information may significantly influence some users' preferences, this study used the artist's name (artist_name) to fill in the missing composer and lyricist fields. This field is highly complete and serves as an effective alternative information source.

For the gender field, "female" and "male" were first encoded as -1 and 1, respectively, while missing values were uniformly filled with 0. Users marked as 0 can be considered as those who prefer not to disclose their gender information, and they themselves constitute a discriminative feature. For other categorical features, missing values are uniformly filled with 0 if the original field does not contain 0, indicating the "unknown" category.

(1) Age field (bd)

Age is an important feature that reflects user preferences. Different age groups often correspond to different musical tastes. There are age anomalies in the original data, such as those over 1000 or negative values, and the proportion of zero values is relatively high. This study regards samples older than 80 or younger than 8 as anomalies and uniformly treats them as 0. Zero-value age is similar to zero-value gender and can be regarded as a behavior of users choosing to hide their personal information. It is retained as a class of features.

(2) User registration time

The subscription duration is calculated by subtracting the user's membership expiration time from the registration time to reflect user loyalty and activity.

(3) Song genre (genre_ids)

Since more than 80% of songs correspond to only a single genre, in order to simplify the feature representation, the first item of the multi-label genre is uniformly taken as the representative category. The same processing method is also applied to the composer and lyricist fields.

(4) Language

As a categorical feature, the language field is converted using One-Hot Encoding.

(5) Song duration (song_length)

The values of this field are concentrated between 2 and 4 minutes, with low discrimination. It is preliminarily judged that its impact on users' listening behavior is limited, so it is deleted.

(6) Artist name (artist_name)

Some songs correspond to multiple singers, and even collective labels such as "group of stars" appear, with up to 27 singers. To reduce the feature dimension and complexity, this study only selects the first singer as a representative. The composer and lyricist fields are treated the same way.

(7) City (city)

Most of the values in this field are 1, the information value is limited, and the correlation with music preferences is weak, so it is deleted. After completing the above processing, the final feature set constructed includes the following dimensions:

- Basic Features:

Song attributes: song_id, genre_ids, artist_name, composer, lyricist, language;

User attributes: msno, bd, gender, registered_via;

Context attributes: source_system_tab, source_screen_name, source_type;

- Cross Features:

msno * genre_ids, msno * source_system_tab, song_id * language;

- Sequence Features:

Combines sequence items and positional embeddings, totaling (6+1) categories;

All basic features total 16 dimensions, providing structured input for subsequent modeling.

5 EXPERIMENTAL RESULTS

In this experiment, we focused on building a recommendation model using a recommendation algorithm based on the BST-Behavior-Sequense-Transformer model. We then compared the results obtained with other methods, including a recommendation algorithm based on the LightGBM decision tree model and a logistic regression approach. This article focuses on the experimental process and analysis of the results using the BST-Behavior-Sequense-Transformer-based recommendation algorithm.

Using the BST-Behavior-Sequense-Transformer recommendation model, we trained the model using 80% of the previously allocated training set, employing a cross-entropy loss. The model's Area Under the Circumference (AUC) curve (which measures a model's ability to distinguish between positive and negative classes) is shown in Figure 11 below:

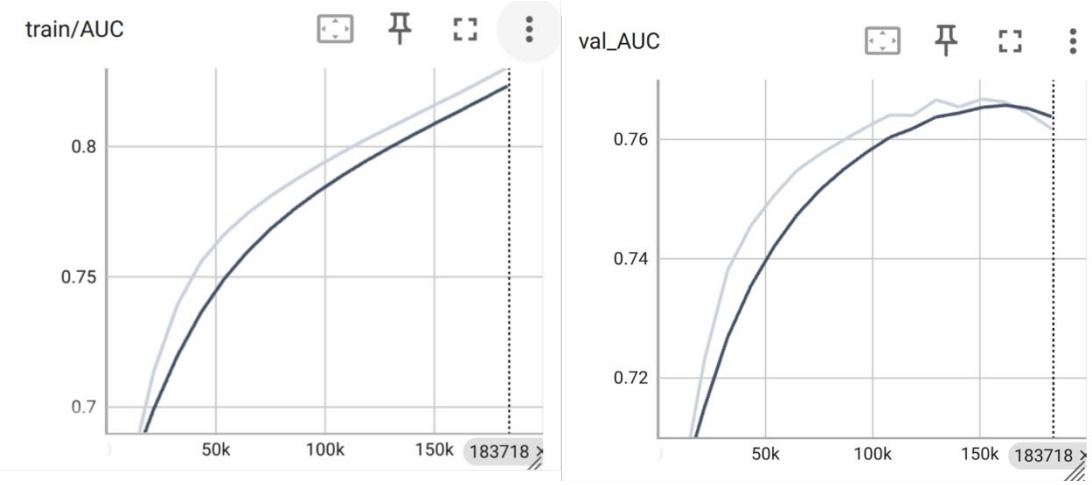

Fig. 11: AUC Curve

As the number of model iterations increases, the AUC value for the model on the train dataset increases, while on the test set it first increases and then decreases, reaching a maximum point.

The model's loss function curve and the root mean square error (RMSE) curve as the number of iterations increases are also shown in Figure 12 below.

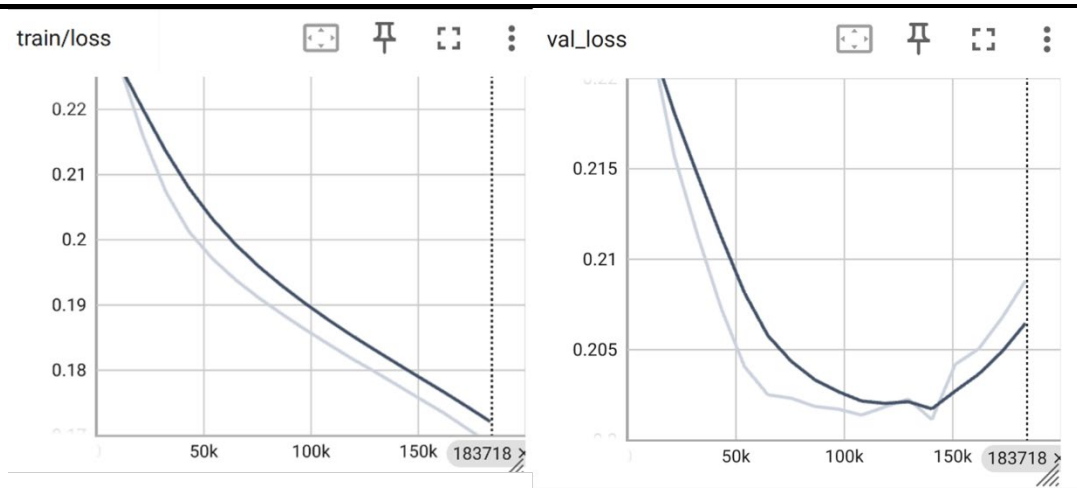

Fig. 12: loss curve

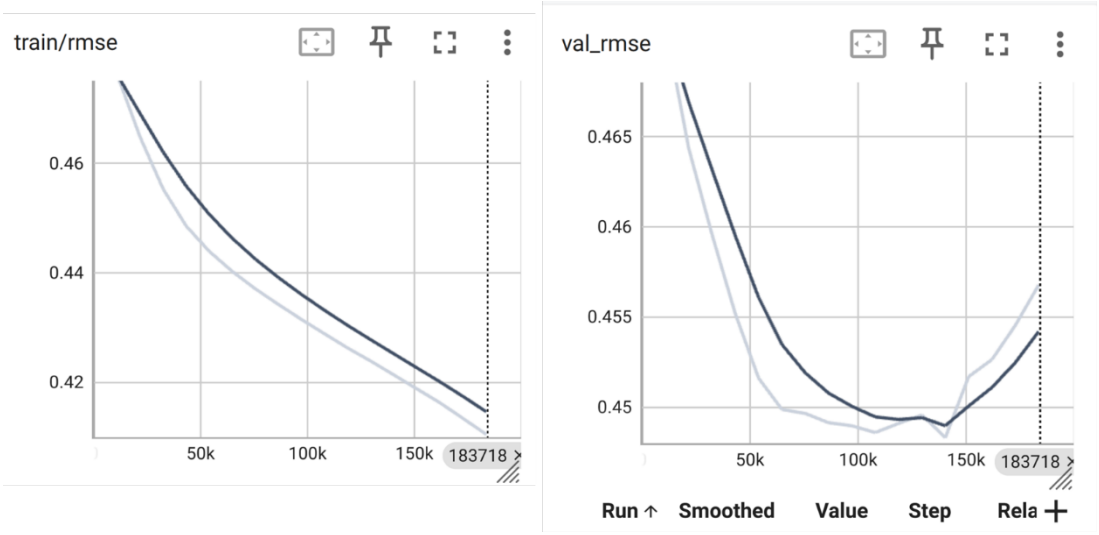

Fig. 13: Root Mean Squared Error (RMSE)

By observing the above curves, we can see that when the model is still underfitting, the AUC value continuously increases with the number of model iterations, while the loss value and root mean squared error (RMSE) continuously decrease, indicating that the model performance is gradually improving. At this point, the number of epochs (Such as, the number of times the model fully traverses the dataset) is less than 12. Training continues. When the model training epochs exceed 12, the model iterations show that the AUC value on the test set begins to decrease, while the loss value and RMSE value begin to increase, indicating that the model has begun to overfit. Therefore, the optimal epoch number for the model is 12.

The AUC value obtained using this model is 0.7662, and the average accuracy reaches 75.51%.

The recommendation model is constructed using the LightGBM boosted tree model algorithm. Again, using the same dataset, the maximum number of trees is set to a variable value, and the tree depth is set to the default 20. After achieving the highest AUC value, the model is iterated for 7 more times. When the AUC value stops improving, the iteration is terminated. The highest AUC value was 0.6579.

```
Starting training...
Training until validation scores don't improve for 7 rounds
Early stopping, best iteration is:
[57]    valid_0's auc: 0.659715
Saving model...
Starting predicting...

0.659715233161362
```

Fig. 14: LightGBM's optimal AUC value.

The accuracy of this algorithm model was 64.69%.

Thus, using a basic logistic regression method to obtain an average accuracy value, we compared the model accuracies obtained by the three methods. The results are shown in Table 5 below.

Table 5: Comparison of Model Performance Under Different Recommendation Algorithms

Recommendation Algorithm	AUC curve value	Accuracy
Logistic Regression	0.6871	67.68%
BST-Behavior-Sequense-Transformer	0.7662	75.51%
LightGBM Boosted Tree	0.6579	64.69%

By comparing the results, we found that the results of the LightGBM boosting tree were not as good as we expected, and were worse than logistic regression. After discussion, we believe that the boosting tree input too many features, resulting in serious overfitting of the model, so its performance on the test set was not very good. However, we are quite satisfied with the feature importance results obtained by the LightGBM boosting tree. The results of the importance of other features obtained by LightGBM are shown in Figure 15 below:

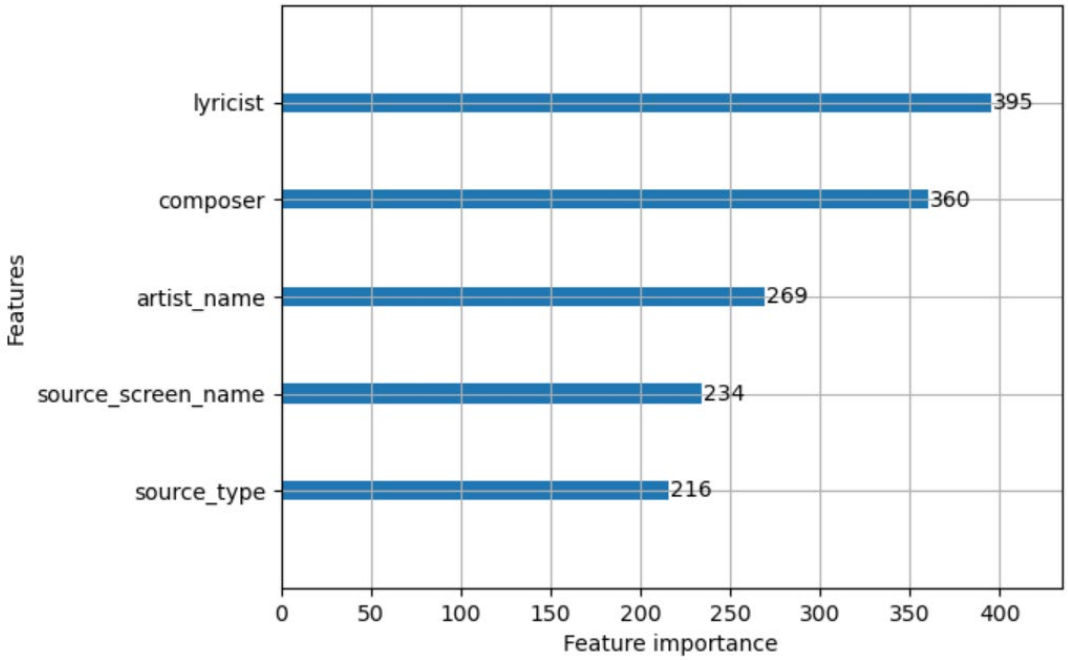

Fig. 15: Feature importance

Online music platforms might consider paying more attention to these feature information to build better personalized music recommendation systems.

As for the other result, the BST-Behavior-Sequense-Transformer recommendation algorithm, which we primarily studied, achieved an accuracy rate of 75.51%, which was quite consistent with expectations. While the results may not be entirely accurate due to some simplification, they are still quite good.

In summary, we chose to use the BST-Behavior-Sequense-Transformer model for user music recommendations, and some of the results are shown below.

Table 6: User Music Recommendation Results

User ID	User age	User gender	Top four recommended songs
28755	0	0	'Flashing' 'Anaesthetic' 'Format' 'Hotel Belfort' 'ปล่อยฉัน'
15402	0	0	'Eminence Front' 'Creatures Of Habit' 'I Swear ' 'Please Don't Say You Love Me'
22313	22	-1	'III. Danse' 'No Slouch' 'Promenade III' 'In Trance'
19542	58	-1	'Le repos de la terre' 'Rest' 'I Am Your Joy and Fury' 'Plo Koon'
16203	46	1	'Ride Like The Wind' 'Pero Tú' 'Want to Want'

6 DISCUSSION

In today's era, with the widespread dissemination and profound development of music culture, more and more users are turning to online music platforms to obtain audio content

that suits their personal preferences. However, as the platform's music library continues to expand, users face problems such as choice overload, shifting interests, and evolving aesthetic demands, increasingly highlighting the limitations of existing music recommendation systems in providing personalized services. Therefore, effectively improving recommendation quality to adapt to users' dynamically changing preferences and enhance user stickiness and platform satisfaction has become a key issue that online music platforms urgently need to address.

Based on research on existing music recommendation algorithms and an analysis of related technological development trends, this paper proposes the following recommendations for optimizing recommendation systems on online music platforms. First, continuous algorithmic innovation should be pursued, such as adopting hybrid recommendation mechanisms that integrate content-based and collaborative filtering approaches to balance the complementary information between music features and user behavior. Specifically, content-based recommendations can match audio features such as style, rhythm, and melody, and are suitable for scenarios where user preferences are clear. Collaborative filtering, on the other hand, generates recommendations for target users by mining historical behavioral data from similar user groups. This is more suitable for users with vague interests or those in the exploratory phase. Furthermore, deep learning models, such as the Behavior Sequential Transformer (BST), can be introduced to more precisely model the complex, nonlinear relationships between user preferences and music items. Furthermore, novelty metrics should be introduced to avoid over-recommendations of content already familiar to users, thereby mitigating the "information cocoon" effect. Multimodal recommendation is also a key direction. By integrating multiple sources of information, such as audio signals, cover images, and lyrics, a more comprehensive understanding of music content can be achieved, improving the accuracy and interpretability of recommendations.

Secondly, the design of user interaction and feedback mechanisms should be strengthened. Platforms can provide explicit feedback interfaces (such as likes, skips, and favorites) to enable users to express their opinions on recommendations in real time, thereby dynamically adjusting recommendation strategies through online learning mechanisms. Introducing social recommendation features, such as integrating friends' playlists or the listening behavior of those they follow, can also help enhance community interactivity and user engagement. Furthermore, systematic user profiles should be constructed. Based on multi-dimensional data such as registration information, historical listening history, and search queries, a structured representation of user preferences should be formed, providing a more reliable basis for personalized recommendations.

Third, the diversification, updating, and maintenance of music content should be continuously promoted. The platform needs to continuously expand its music library to cover diverse genres, styles, and eras to meet the needs of different user groups. Establishing popular charts and themed playlists can effectively guide users to discover new and popular content. Furthermore, it should actively develop personalized playlist generation mechanisms that automatically assemble music sequences that match users' preferences based on their real-time and historical behavior, enhancing the listening experience.

Fourth, scenario-based recommendation functions can be expanded. By establishing dedicated music sections for specific scenarios, such as exercise, sleep, study, and travel, and integrating the user's current environment, time, and behavioral context, intelligent scenario

matching and music push can be implemented to make recommendations more tailored to users' actual needs.

Finally, a comprehensive data analysis and monitoring system should be established. Regularly collect and analyze user behavior data to identify shifts in preferences and popular trends, supporting iterative optimization of the recommendation model. Furthermore, key metrics of recommendation effectiveness, such as click-through rate, complete play rate, and user satisfaction, should be monitored in real time, and algorithm strategies should be rapidly adjusted based on feedback data to maintain system efficiency and adaptability.

7 CONCLUSIONS

This study systematically compares the performance of different recommendation algorithms on music recommendation tasks, aiming to construct an optimal recommendation model and provide practical system optimization recommendations for online music platforms. The core contribution of this paper lies in the introduction of a user behavior sequence modeling method based on the attention mechanism and the Transformer architecture (Such as the BST model). This method breaks through the limitations of traditional recommendation algorithms (Such as content-based filtering, collaborative filtering, and rule-based recommendation), demonstrating strong theoretical innovation and practical application potential. However, the current research still has several limitations, and future research is needed to further explore algorithmic diversity, feature engineering, and system evaluation.

First, this study conducted limited but representative exploration at the algorithmic level. While focusing on state-of-the-art deep learning models (Such as the BST algorithm and LightGBM), it does not include more emerging recommendation technologies such as multimodal learning, reinforcement learning, or knowledge graph-based recommendation methods. Furthermore, the feature construction and selection process still relies heavily on artificial priors and does not fully utilize feature importance analysis (Such as evaluating feature contributions through LightGBM) to optimize high-dimensional feature inputs. This may result in the model failing to fully capture the complex relationship between user preferences and music content. Future research could introduce automated feature engineering (AutoFE) or graph neural network-based feature representation methods to further enhance model expressiveness.

Secondly, data size and quality significantly impact model performance. The dataset used in this study contains over seven million samples, encompassing multi-dimensional user behavior and music metadata. However, data cleaning, integration, and enhancement still present significant challenges. This is particularly true when using external data sources (such as KKbox), which present issues such as field redundancy, inconsistent annotations, and sparsity. Future work could consider introducing generative adversarial networks (GANs) or self-supervised learning strategies to expand training samples and improve the robustness of data representation.

Finally, the algorithm evaluation criteria are relatively limited, failing to fully encompass key metrics such as user experience, interpretability, bias, and fairness. Music recommendation systems must not only pursue accuracy but also consider diversity, novelty, and contextual adaptability (Such as users' real-time emotions, device environment, and cultural background).

Furthermore, this study did not delve into the "information cocoon" effect and its mitigation mechanisms. Future work could incorporate the exploration-exploitation framework from reinforcement learning to dynamically balance the relationship between familiar content and the discovery of potential interests. In summary, while this research has made initial progress in behavioral sequence modeling and deep learning recommendation techniques, personalized music recommendation remains a challenging, multidisciplinary field. Further work is needed in four areas: algorithm innovation, data governance, evaluation systems, and human-computer interaction design, to drive the next generation of music recommendation systems towards smarter and more user-friendly features.

ACKNOWLEDGEMENTS

This work was supported by Jiangxi Provincial College Students' Innovation and Entrepreneurship Program (S202510421047).

REFERENCES

[1] Zhang, Z., Qiu, T., Liao, Z., Liu, H., Wei, C., & Zhao, T. (2025, July). Research on global power equipment trade prediction and logistics optimization based on 5G and edge computing. In International Conference on Optoelectronics and Information Technology (ICOIT 2025) (Vol. 13665, pp. 302-310). SPIE.

[2] Zhao, T., Chen, G., Pang, C., & Busababodhin, P. (2025). Application and Performance Optimization of SLHS-TCN-XGBoost Model in Power Demand Forecasting. CMES-Computer Modeling in Engineering and Sciences, 143(3), 2883-2917.

[3] Zhao, T., Chen, G., Suraphee, S., Phoophiwfa, T., & Busababodhin, P. (2025). A hybrid TCN-XGBoost model for agricultural product market price forecasting. PLoS One, 20(5), e0322496.

[4] Wang, Z., Zhang, Z., Su, T., Ding, Z., & Zhao, T. (2024, December). Research on Supply Chain Network Optimisation Based on the CNNs-BiLSTM Model. In 2024 International Conference on Information Technology, Comunication Ecosystem and Management (ITCEM) (pp. 197-202). IEEE.

[5] Zhao, T., Chen, G., Gatewongsa, T., & Busababodhin, P. (2025). Forecasting Agricultural Trade Based on TCN-LightGBM Models: A Data-Driven Decision. Research on World Agricultural Economy, 207-221.

[6] Akkaya, B. (2025). Current Trends in Recommender Systems: A Survey of Approaches and Future Directions. *Computer Science*, *10*(1), 53-91.

[7] Yang, B., Zhou, J., Zhang, S., Xing, Y., Jiang, W., & Xu, L. (2024). Lightweight knowledge distillation and feature compression model for user click-through rates prediction in edge computing scenarios. *IEEE Internet of Things Journal*.

[8] Liang, W., Meo, P. D., Tang, Y., & Zhu, J. (2024). A survey of multi-modal knowledge graphs: Technologies and trends. *ACM Computing Surveys*, *56*(11), 1-41.

[9] Chang, Y. J., Han, J. Y., Chu, W. C., Li, L. P. H., & Lai, Y. H. (2024). Enhancing music recognition using deep learning-powered source separation technology for cochlear implant users. *The Journal of the Acoustical Society of America*, *155*(3), 1694-1703.

[10] Tao, S., Qiu, R., Cao, Y., Xue, G., & Ping, Y. (2023). Path-guided intelligent switching over knowledge graphs with deep reinforcement learning for recommendation. *Complex & Intelligent Systems*, *9*(6), 7305-7319.

[11] Zhang, M., Liu, H., Chen, C., Gao, G., Li, H., & Zhao, J. (2023). AccessFixer: Enhancing GUI accessibility for low vision users with R-GCN model. *IEEE transactions on software engineering*, *50*(2), 173-189.