Hybrid Book Recommendation System Using Collaborative Filtering and Embedding Based Deep Learning

Ouahiba Remadnia, Faiz Maazouzi, Djalel Chefrour
Dept. of Computer Science University of Souk Ahras, Algeria
E-mail: w.remadnia@univ-soukahras.dz, f.maazouzi@univ-soukahras.dz, djalel.chefrour@univ-soukahras.dz

Keywords: Book recommendation system, collaborative filtering, hybrid architecture, deep learning, embedding layer, e-learning application, online education.

Received: August 20, 2024

We propose a hybrid e-book recommendation mechanism that leverages collaborative filtering and content-based recommendation paradigms to address inherent challenges in e-learning systems. For collaborative filtering, we present an innovative deep learning framework that utilizes embeddings to enhance accuracy and manage large datasets efficiently. This framework effectively addresses the cold start problem, thereby improving recommendation precision. In content-based recommendation, we introduce a regression-based technique to elevate system capabilities by incorporating content attributes. The integration of these techniques into our deep learning model creates a comprehensive and adaptable solution with scalability and effectiveness. Experiments on the Book Recommendation dataset demonstrate that our solution provides better suggestions and outperforms existing works in terms of Root Mean Square Error (RMSE) and Mean Absolute Error (MAE), achieving values of 0.69 and 0.51, respectively.

Povzetek: Predstavljen je hibridni sistem priporočil knjig, ki združuje sodelovalno filtriranje in globoko učenje z vgrajenim slojem in učinkovito rešuje težave hladnega zagona.

1 Introduction

Over the past few years, the popularity of artificial intelligence (AI) has increased dramatically, leading many services to rely heavily on it. Society has become increasingly dependent on electronics and AI, with numerous tasks and achievements being accomplished through its use [11] .Among the applications of AI is its use in education [19] , where its impact is amplified by recommendation systems (RS), which have become an integral part of our daily lives. These systems act as a logical first line of defense against excessive consumer choice. Generally, these systems generate a list of suggestions based on the user's profile and behavior, including their interaction with the available offers, item features, and other relevant information.

However, unlike search engines or retrieval systems that provide relevant results based on the user's queries, RS offer suggestions specifically tailored to the user's needs and preferences [9], [16]. They are critically important in sectors such as e-commerce , tourism [28], and online video platforms. Examples of real-world RS include Amazon's and Netflix's personalized recommendations for books and movies.

Particularly, and in addition to the aforementioned fields, researchers have paid more attention to the area of e-learning RS [10], [31], [30], [21], focusing on developing cutting-edge methods for tailoring recommendations to each learner's specific requirements. Individuals face a significant challenge in sorting through massive amounts of data to locate the information they need as the availability of e-learning applications continues to grow. Adaptive e-learning and other forms of personalized technology, like RS, have evolved as solutions to this problem.

On the other hand, recent embedding layer technology-based RS have made it possible to use a wider variety of information to forecast user preferences by incorporating data about the user and the item. As part of our study, we decided to conduct an in-depth inquiry into the difficulties related to book RS, which are an essential component of online education. These systems have demonstrated tremendous utility in a wide variety of educational settings [8], such as classrooms, libraries, and online instructional websites, among others. Reading content has become much simpler and more convenient for readers as a result of the broad availability of electronic books and their affordable prices. As a direct consequence of this, there has been a commensurate increase in the number of people reading printed literature. However, due to the large number of books currently on the market, the use of RS has become an unavoidable necessity.

There is an established need to use RS in the current book industry to guide readers in selecting titles according to their preferences and similarities with other users. This need is justified by the vast collection of books available. RS not only make books easier to find but also encourage readers to explore new literary genres and authors they might not be familiar with. By integrating RS into educational environments, schools can now provide students with

individualized reading recommendations that supplement their coursework and foster a lifelong passion for learning.

There are currently three main research directions in the field of RS: content-based recommendation [4] [29], collaborative filtering-based recommendation [15] [33], and hybrid recommendation methods [13].

Our motivation stems from the need to create a RS that not only excel in accuracy but also adapts to the dynamic nature of user preferences and the diverse attributes of books. By combining collaborative filtering and content-based methods within a deep learning framework, we aim to develop a robust and scalable solution that enhances the user experience and fosters deeper engagement with reading materials.

Our research is centered on gaining a comprehensive understanding of book RS . The methodologies we adopted are hybrid approaches that combine content-based and collaborative filtering techniques. Additionally, as a novel aspect of our work, we exploit the recent embedding layer technique [17] [18], which employs a wider variety of information to forecast user preferences by including data about the learner and the book. This proposed system provides customized suggestions that go beyond just the most popular titles by evaluating the learner's behavior as well as their reading patterns and feedback. This enables readers to explore a diverse selection of books that align with their individual interests, enhancing their reading experience and allowing them to discover new, compelling content.

Moreover, our RS address crucial points such as ethical and moral responsibilities towards its users. Specifically, it takes measures to protect user privacy, address the potential for algorithmic bias, and establish reliable evaluation metrics. By implementing preventative measures to tackle these challenges, we aim to fully capitalize on the promise of recommendation systems to enhance the reading experience for everyone. The empirical evaluation of our RS demonstrates that it outperforms similar existing works in terms of Root Mean Square Error (RMSE), with a value of 0.69, and Mean Absolute Error (MAE) of 0.51.

The rest of this paper is organized as follows: We begin with an introduction in Section 1 . Then, in Section 2, we review the related works. After that, in Section 3, we present the general scheme of our work and discuss our contributions. Next, in Section 4, we demonstrate our dataset and preprocessing. Following that, in Section 5, we discuss the proposed model and its architecture. Later, in Section 6, we present the experimental results. In Section 7, we compare our results with existing approaches. Finally, we conclude in Section 8 and outline potential future works.

2 Related work

The importance of RS has grown with the rise of user-generated data, prompting more research and the development of innovative solutions to help users manage the overwhelming number of options. This paper covers different types of RS, the embedding layer, and explores how RS is applied in various fields, with a focus on e-learning.

2.1 Techniques used for recommendations

When it comes to RS, there are three main approaches that are commonly used: collaborative filtering (CF), content-based recommendation (CB), and hybrid RS.

2.1.1 Collaborative filtering recommendation systems

CF is a type of RS that relies on the behavior and preferences of a group of users to provide recommendations to individual users [32]. It works by identifying patterns of similarity and difference between users and their interactions with items or content. These patterns are then used to generate recommendations for users based on the behavior of similar users.

An example of CF in e-learning is when an online course platform suggests additional courses to a user based on the course selections and behavior of other users who have similar preferences or learning paths. For instance, if a user takes a course in data analysis, the system can use collaborative filtering to recommend other data-related courses to that user based on the behavior of others who have taken similar courses. This can help personalize the learning experience and make it more relevant to the user's interests and goals [24].

2.1.2 Content-based recommendation systems

On the other hand, content-based RS rely on item features and metadata to make recommendations. CB systems are useful for recommending items that are similar in content or style to those that the user has already shown interest in. This is achieved by calculating the degree of similarity between the various features associated with each item. For example, if a user enjoys a particular comedy movie, the system can use that information to recommend other movies in the same genre [24]. Figure 1 explains CF and CB filtering.

2.1.3 Hybrid recommendation systems

By combining elements of both CF and CB approaches, we can create a more comprehensive and personalized RS. This hybrid model leverages both user behavior and item features. An example of a hybrid model in e-learning is when an online course platform uses a combination of CF and CB filtering to provide course recommendations to users. The system can utilize CF to identify similar users who have taken analogous courses and then employ CB filtering to recommend courses that align with the user's learning style, goals, and interests. This approach can yield more accurate and personalized recommendations, as it considers both user behavior and item characteristics [5].

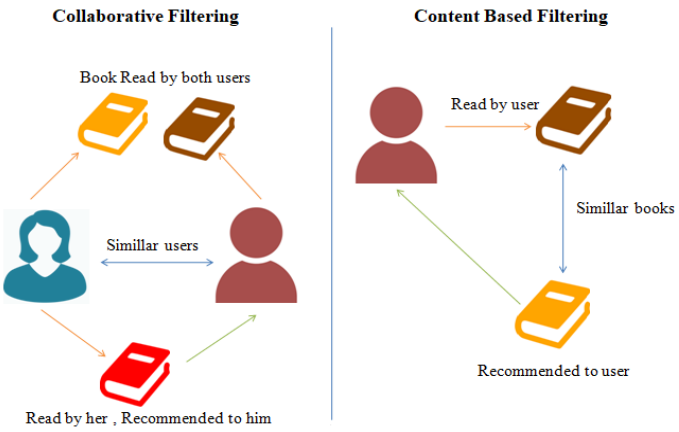

Figure 1: Collaborative filtering and content based filtering

2.2 Embedding layer

The embedding layer is a component within a neural network that transforms categorical data into continuous, dense vectors of a predetermined size. The primary objective of the embedding layer is to obtain a comprehensive representation of the categorical input that is well-suited for use in machine learning models, particularly neural networks. In research studies employing deep learning, it is common practice to utilize an embedding layer to convert categorical data into continuous vectors, which can then be input into a neural network [22].

The figure below shows the architecture of the embedding layer.

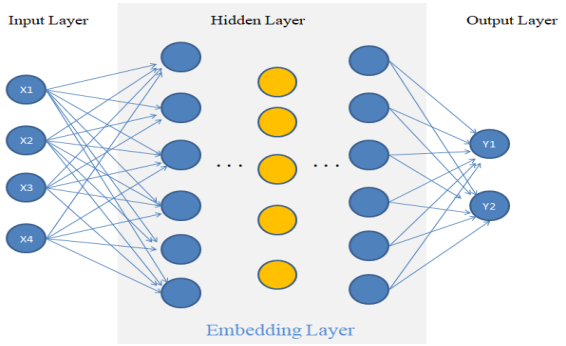

Figure 2: Model of embedding layer

Although embedding layers were initially developed for natural language processing (NLP) tasks, the concept quickly spread to other domains with sparse input vectors. Most notably, in RS, the user-item matrix is typically a very sparse matrix containing a large number of zeros. Some vanilla implementations, such as item2vec by O. Barkan et al. [3], use the same embedding layers but, instead of words, use products. They represent each user as a product vector and use embedding layers to find users that are close to one another in this product space or, similarly, to find embeddings of products that are close to one another in the user space.

Additionally, Neural Personalized Embedding [18] by T. Nguyen et al. attempts to extend traditional matrix factorization recommenders by incorporating item embeddings to address cold start issues. A. Damian et al. [6] present a practical implementation of this procedure for pharmaceutical retail recommenders.

2.3 Relevant work

We aim to review studies on RSs and their application in helping online learners navigate large amounts of data to find training materials. Adaptive e-learning and RS provide effective solutions to improve learners' access to relevant resources. In their work, R. Anand and J. Beel [2] introduce Auto-Surprise, an automated recommender system library that optimizes algorithm selection and configuration, outperforming the original Surprise library. Their results demonstrate improved performance and faster hyperparameter tuning across various datasets, highlighting the potential for automation to enhance RS effectiveness.

Similarly, Tarus et al. [27] developed a hybrid RS for e-learning that incorporates context awareness and sequential pattern mining (SPM) approaches. By leveraging contextual information and the learner's knowledge, their system improves recommendations and addresses common challenges such as data shortage and cold start issues. Experimental results show that this approach enhances the effectiveness of the e-learning platform, further underscoring the role of advanced techniques in improving RS performance across different domains.

Additionally, Porcel et al. [20] presented a fuzzy linguistic RS for digital libraries, where selective diffusion removes irrelevant items and displays only the most important ones. Testing shows that the fuzzy linguistic RS outperforms traditional systems in terms of accuracy, diversity, and originality, providing a more customized and user-friendly experience.

Moreover, the enormous number of available books makes the use of RS a necessity. Several studies combine conventional CF with CB methods to enhance the quality of recommendations. For example, Rajpurkar et al. [23] integrated books suggested by a content-based RS into an item-based CF approach.

Simultaneously, association rules are identified, and the system retrieves books that appear in the same transaction as the user's preferred products. The system suggests recommendations based on the intersection of CF results and association principles.

Jomsri [12] designed the FUCL paradigm to enhance library services. The model recommends books to users based on their library borrowing history and academic background. ARM was used to generate these recommendations. The system's efficacy was evaluated using precision and recall metrics, demonstrating greater precision than other techniques.

Ali et al. [1] retrieved book tables of contents and stored metadata offline.

When a new user orders a book, their profile is collected and stored, and association rule mining (ARM) is used to match books to user preferences and make recommendations. The web-based solution performed well using precision, recall, and F-measure criteria.

Zhang et al. [34] utilized Chinese library classification to recommend books in Chinese libraries. The system makes personalized recommendations based on user choices, outperforming typical recommendation approaches in precision, recall, and F-measure.

The table below provides a comparative analysis of various research papers on RS.

2.4 Research gap

The previous review provided valuable insights into several practical limitations inherent in existing book recommendation systems, despite their significant role in the e-learning domain, a field that remains somewhat overlooked in comparison to the more widely studied areas of modern research. Specifically, these systems face challenges such as the cold start problem, where recommendations for new users or books are limited, and the difficulty in handling sparse data, which leads to incomplete or inaccurate suggestions. Additionally, personalization remains a major hurdle, as these systems often fail to capture the complex relationships between users and books, limiting their ability to provide truly tailored recommendations. Furthermore, the lack of recognition of subtle patterns within the data exacerbates these issues, reducing the system's overall effectiveness and its potential to offer more meaningful, context-aware suggestions for users.

More specifically, the research works reviewed present several limitations across different approaches to recommender systems. The Auto-Surprise library faces challenges in selecting the best algorithms and hyperparameters due to the sensitivity of performance to minor variations in implementation and parameter settings. Fuzzy linguistic recommender systems encounter difficulties in managing qualitative information effectively, particularly when dealing with diverse linguistic granularities in digital libraries. The book RS by Sushama Rajpurkar et al. struggles with CB filtering inability to distinguish between high- quality and low-quality content if similar terminology is used. Zafar Ali et al. hybrid book RS highlights that existing recommenders often fail to conduct deeper content analysis, focusing mainly on surface-level descriptions and metadata. Additionally, the CF system by Khishigsuren Davagdorj et al. faces the typical issue of sparsity, where users do not rate all items, leading to incomplete data matrices. Lastly, the proposed approach for book recommendation based on User k-NN also grapples with challenges such as cold start problems and data sparsity inherent in collaborative filtering methods. These limitations underscore the need for more robust and adaptable solutions in the development of recommender systems. By contrast, our work addresses some of these problems with a novel approach, as explained in our contributions list in the following section.

3 General architecture and contribution

3.1 General architecture

The following diagram illustrates the overall architecture and key components of the hybrid RS we developed.
The general architecture of our hybrid RS is depicted in Figure 3, which shows how the system combines CF and CB filtering with the use of deep learning embeddings to improve recommendations. Here's a step-by-step explanation of the image and the process it represents:

1. The system collects in a dataset the information about users, books, and the ratings that users have given to those books.
2. Feature extraction occurs, where the users and books data are represented as vectors.
3. On the one hand, with CB filtering, the system recommends books by analyzing the characteristics of new users using their profiles. It employs regression techniques to predict user preferences and provide suggestions, effectively addressing problems like the cold start issue.
4. On the other hand, the system employs CF to focuse on learning user preferences based on the preferences of other users with similar tastes. It identifies patterns in user behavior and ratings to recommend books that users with comparable profiles have liked. An embedding layer is a key component of the system, enabling efficient representation of user and book information.
5. The hybridization step combines CF and CB filtering results to offer more accurate and personalized book recommendations.
6. Finally, the system ranks books based on predicted ratings and recommends the top-ranked ones to each user, resulting in a personalized list of book recommendations.

3.2 Contribution

This paper describes a neural network model specifically developed for recommending books to users based on their preferences. Our method employs CF, allowing the model to predict a user's preferences by leveraging the preferences of other users with similar interests.

The primary contribution of our work lies in the utilization of embedding layers, which effectively represent users and books in a high-dimensional space. Embeddings are a

Table 1: Comparative table of recommender system research papers

REF	Year	Contributions	Algorithms	Datasets	Measures
[7]	2020	A sophisticated library designed to automate the selection and configuration of algorithms for recommendation systems (RS).	Auto-Surprise (TPE), Auto-Surprise (ATPE)	Book Crossing dataset	RMSE: 0.70, MAE: 0.45
[26]	2019	Proposed a book recommendation system using collaborative filtering with user k-NN.	User k-NN, Pearson similarity, Cosine similarity	Book Crossing dataset	RMSE: 2.99, MAE: 2.63, NMAE: 0.29
[34]	2017	Enhanced accuracy and personalization in book recommendation systems through the application of hierarchical classification techniques, resulting in more tailored and relevant suggestions for individual users.	(ULLRM), (DRFM)	University library book data	Precision: 0.61, Recall: 0.72, F-Value: 0.66
[23]	2015	Enhanced effectiveness of digital library recommendations with association rules.	Association Rule Mining	Records of book loans from digital libraries	Support, Confidence of association rule
[27]	2018	Provided personalized learning recommendations by incorporating behavior patterns and contextual data.	Sequential Pattern Mining, Context-Aware Recommendations	E-learning platform user interaction data	F1-measure: 0.32, MAE: 0.75
[20]	2017	Development of fuzzy linguistic recommender systems for selective information dissemination in digital libraries.	Fuzzy linguistic modeling, Multi-granular linguistic information, Hybrid recommendation approach	Various digital library datasets	User retention, System performance
[1]	2016	Development of a hybrid book recommender system using TOC and association rule mining.	Content-based filtering (CB), Collaborative filtering (CF), Association rule mining	Locally available university course-related books	Precision: 0.79, Recall: 0.74, F-measure: 0.76
[2]	2020	Development of Auto-Surprise, an advanced automated recommender system library integrated with Tree-structured Parzen Estimators (TPE) optimization.	Tree of Parzens Estimator (TPE), Adaptive TPE (ATPE)	Book Crossing	RMSE: 3.52, MAE: 2.88

powerful technique capable of capturing intricate relationships between variables, with proven effectiveness across various domains such as NLP and computer vision. In our model, we employ backpropagation during training to learn these embeddings, enabling the model to uncover nuanced patterns in the data that conventional feature engineering techniques often miss.

Additionally, our approach incorporates bias terms to account for individual differences in user and book preferences. These bias terms provide a simple yet effective means of enhancing the accuracy of CF models by considering factors such as book popularity and user-specific preferences, allowing for the adjustment of book ratings above or below average.

The use of embedding layers addresses the sparsity problem common in recommendation systems. Since users typically rate only a small subset of available items, embeddings help to identify latent characteristics and similarities between users and books, even when direct interactions are sparse.

Moreover, incorporating bias terms effectively tackles the challenge of personalized recommendations. By taking into account the unique preferences and idiosyncrasies of each user, the model generates recommendations that align closely with individual tastes.

Overall, the integration of CF, embedding layers, and bias terms significantly enhances the precision and effectiveness of our RS. These innovations address critical chal-

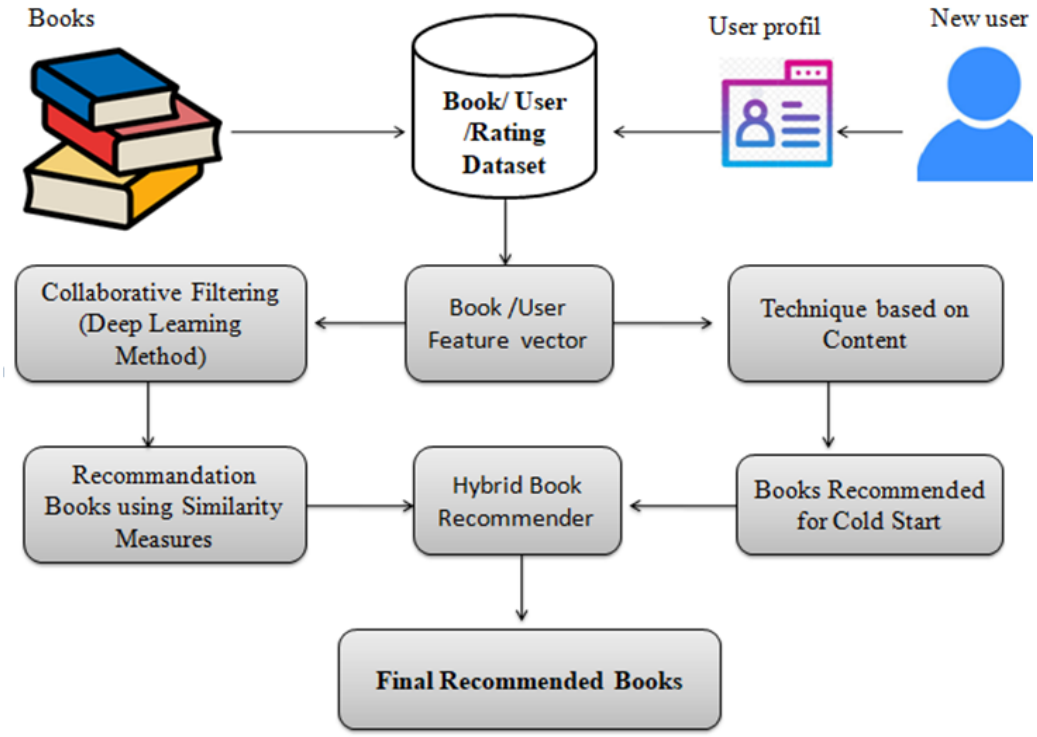

Figure 3: General architecture

lenges such as the cold start problem, sparse user-item interactions, and the need for personalized recommendations. The effectiveness of our approach is demonstrated by the outstanding performance of our model on benchmark datasets, achieving RMSE and MAE values of 0.69 and 0.51, respectively. These results highlight the efficiency and significance of our proposed hybrid approach. Our work effectively addresses key challenges in recommendation systems, such as personalization difficulties, cold start problems, and sparse data handling. Unlike many existing methods, our approach integrates collaborative filtering with advanced deep learning techniques, particularly embeddings, to significantly enhance the accuracy of user preference predictions.

4 Dataset

4.1 Dataset description

For our project, we used the Book Recommendation Dataset from Kaggle. We use The Book Recommendation Dataset [35]. which is a widely recognized dataset for developing and evaluating book recommendation systems. This dataset was selected due to its comprehensive collection of user ratings, book information, and user demographics. It provides a rich foundation for training models that can capture complex patterns in user preferences. The dataset includes three main files: Books.csv, Users.csv, and Ratings.csv, The dataset includes over 1.1 million ratings from over 53,000 users on more than 27,000 books.

– Data Splitting:

  The dataset was split into training (80%) and testing (20%) subsets to create a robust foundation for learning, validation, and evaluation, we could clarify why the 80-20 split was chosen—this is typically a standard practice to ensure enough data for training while preserving enough unseen data for evaluation.

– Choice of Dataset:

  The Kaggle Book Recommendation Dataset was chosen because it provides a large and rich set of data regarding books and users, making it ideal for building and evaluating a recommendation system.

4.2 Preprocessing of a dataset

Essential steps such as data cleaning, removal of irrelevant attributes, merging datasets, and handling missing values (e.g., imputing mean values for missing ages) were meticulously carried out to prepare and optimize the dataset for training the recommendation model. These processes are aimed at ensuring that the data used is not only complete but also consistent and ready for efficient use in machine learning. The attention given to these steps helps prevent any distortion in the model's results and ensures high data quality, which is crucial for generating accurate and relevant recommendations for users.

– The initial phase, data cleaning, entailed a surgical removal of extraneous attributes that held no relevance to our objectives. Within the realm of our book recommendation dataset, redundant columns such as image URLs were promptly excised, streamlining the data for focused analysis.

– To gain holistic insights, we executed data merging. This unified the diverse datasets into a cohesive entity, enhancing the feasibility of analysis and model integration while avoiding duplication.

– Handling missing values constituted a pivotal facet of our preprocessing journey. Scrutiny of the dataset revealed incongruities within the 'Age' column, encompassing NaN entries and anomalously high values. Addressing these anomalies, ages below 5 and above 110, deemed implausible, were systematically replaced with NaNs. The subsequent step encompassed imputing the miss ing values with the mean age value, subsequently cast as integers to refine data uniformity.

– Factorizing values refers to the process of converting categorical data into numerical form. Specifically, it transforms the categorical values in the Location column into unique integer codes using the factorize() function from the pandas library. Each distinct location is assigned a unique integer, replacing the original string values with numeric codes. This step is essential for preparing categorical data for machine learning models, which require numerical input.

– After factorization, a dictionary is created to map these numeric codes back to their original string values, allowing us to retrieve the data by their real values when needed. This ensures that while the model uses numerical representations, we can still interpret and understand the results in terms of the original categories.

Our efforts in preprocessing have led to the creation of a refined and organized dataset ready for the challenges of recommendation system modeling. The following image illustrates the dataset post-processing.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 840288 entries, 0 to 1149771
Data columns (total 4 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   Location     840288 non-null  category
 1   Age          840288 non-null  float64
 2   ISBN         840288 non-null  category
 3   Book-Rating  840288 non-null  int64
dtypes: category(2), float64(1), int64(1)
memory usage: 34.9 MB
```

Figure 4: Dataset post processing

5 Proposed model

5.1 Collaborative filtering

As a crucial step in developing a robust collaborative filtering (CF) model, our approach centers on filtering data based on users who have assigned similar ratings to a shared selection of books. This strategic curation not only establishes meaningful connections between users but also enhances the effectiveness of recommending highly-rated books within these linked user groups. By leveraging deep learning methodologies, our recommendation system (RS) maximizes personalization, providing users with tailored book suggestions that reflect their past preferences. Importantly, our strategy utilizes the book recommendation dataset as a solid foundation for both constructing and validating the recommendation model, ensuring its reliability and effectiveness in real-world applications.

The architectural foundation of our approach is rooted in a deep learning structure fortified by an embedding layer. By assimilating information from the dataset, our model is primed to discern intricate patterns and correlations. This process is fueled by training the model on 80% of the dataset, with the remaining 20% reserved for meticulous evaluation.

A key aspect of our approach is the incorporation of embedding layers, which play a pivotal role in crafting a compact yet comprehensive representation of users and books as dense vectors of real numbers. The dimensionality of these vectors is tailored to specific requirements, adding a layer of adaptability to the model.

The model further integrates biases, meticulously accounting for both user and book preferences. This dynamic aspect facilitates the fine-tuning of ratings, tailoring recommendations to each individual user's tendencies.

In summary, our proposed model highlights the remarkable effectiveness of CF within RS. By seamlessly interlinking users and books through a carefully designed data filtering process, we set the foundation for an advanced recommendation model powered by deep learning techniques. This complex and sophisticated structure, bolstered by carefully crafted embeddings and enriched with strategically introduced biases, orchestrates a highly personalized experience. It generates recommendations that not only reflect but also adapt to each user's distinct literary preferences and evolving tastes, ensuring a truly individualized and meaningful interaction with the system.

5.1.1 Clarification of bias terms in collaborative filtering

We provide hereafter a precise description of the biases used in our system, how they are calculated, and their roles in the overall prediction. Specifically, we will clarify the following:

– User Bias:

This term reflects the tendency of a particular user to rate items higher or lower than the average rating. It is calculated by taking the average difference between a user's ratings and the overall mean rating across all items. This bias is added to the predicted score to adjust for individual user preferences.

– Item Bias:

This term accounts for the inherent popularity or quality of an item, independent of user ratings. It is computed as the average difference between the item's ratings and the overall mean rating. Similar to user bias, item bias is included in the prediction score to enhance the model's accuracy.

To better understand how user and item biases are integrated into the final prediction score within a collaborative filtering approach, we can examine the following equation, which demonstrates the systematic incorporation of these biases. This process accounts for individual user preferences and item characteristics, adjusting the predicted ratings accordingly to provide more personalized recommendations.

$$\hat{r}_{ui} = \mu + b_u + b_i + \langle p_u, q_i \rangle$$

Where:

– $\hat{r}_{ui}$: Predicted rating of user u for item i

– μ: Global average rating (mean rating across all users and items)

– b_u: User bias term for user u (how much this user's ratings deviate from the average)

– b_i: Item bias term for item i (how much this item's ratings deviate from the average)

– $\langle p_u, q_i \rangle$: Dot product of the user latent factor vector p_u and the item latent factor vector q_i (captures the interaction between user u and item i)

5.1.2 Model architecture

The architecture of the model is composed of two distinct embedding layers: one dedicated to users and the other to books. These embedding layers serve to map each individual user and each book to a unique vector representation in a high-dimensional space, allowing the model to capture complex relationships and preferences. The vectors generated by these layers are integral to the recommendation process, as they enable the model to make personalized predictions. The image above provides a visual representation of our training model, illustrating the flow and interaction between these components to achieve the desired outcomes in recommendation tasks.

In the Dot Product layer, the model performs the dot product operation between the user and book embedding vectors to predict a rating. The dot product is a fundamental mathematical operation used in vector space models, where it takes two vectors of the same dimension and computes the sum of the products of their corresponding components. Specifically, for two vectors uu and vv of dimension nn, the dot product is calculated as the sum of the individual products of their components, as defined by the following formula:

$$dot\_product = u[1]*v[1]+u[2]*v[2]+...+u[n]*v[n] \quad (1)$$

In other words, the dot product quantifies the similarity between two vectors. In the context of book recommendations, this means that the model assesses how closely the user's preferences align with the book's characteristics.

Thus, the Dot Product layer is utilized to learn a representation of the user and the book, capturing their relationship within a shared feature space. To ensure that the predicted rating falls between 0 and 1, the sigmoid function is applied to the output. During training, the model optimizes the embedding vectors and bias terms to minimize the loss function effectively.

The minimization in equation (1) is performed over the embedding vectors u and v and the bias terms, where the objective is to find the values that minimize the difference between the predicted ratings and the actual ratings in the dataset.

The main contribution of this work is that it provides a Let R be the set of user-book ratings, where each rating r is a tuple (u, b, r), representing the rating of user u for book b with value r. The goal of the method is to learn a function f that maps each rating (u, b) to a predicted rating $\hat{r}$.The function f is defined as follows:

$$f(u, b) = \sigma(\langle u, b \rangle + u_{bias} + b_{bias}) \quad (2)$$

where u is the embedding vector for user u, b is the embedding vector for book b, u_{bias} is the user bias term for user u, b_{bias} is the book bias term for book b, $\langle u, b \rangle$ is the dot product of u and b, and σ is the sigmoid activation function.

The embeddings u and b are learned during the training process by minimizing the following loss function. This process involves optimizing the model parameters to effectively capture the underlying patterns in the data.

$$L = \sum_{(u,b,r) \in R} (r - \hat{r})^2 + \lambda_u ||u||^2 + \lambda_b ||b||^2 \quad (3)$$

where r is the actual rating for (u, b), $\hat{r}$ is the predicted rating for (u, b), λ_u and λ_b are regularization parameters to prevent overfitting, and $||u||$ and $||b||$ are the L2 norms of the embedding vectors.

The loss function is minimized through the use of stochastic gradient descent (SGD), an optimization technique that iteratively adjusts the model parameters. During each iteration, the parameters are updated in the direction of the negative gradient of the loss function with respect to

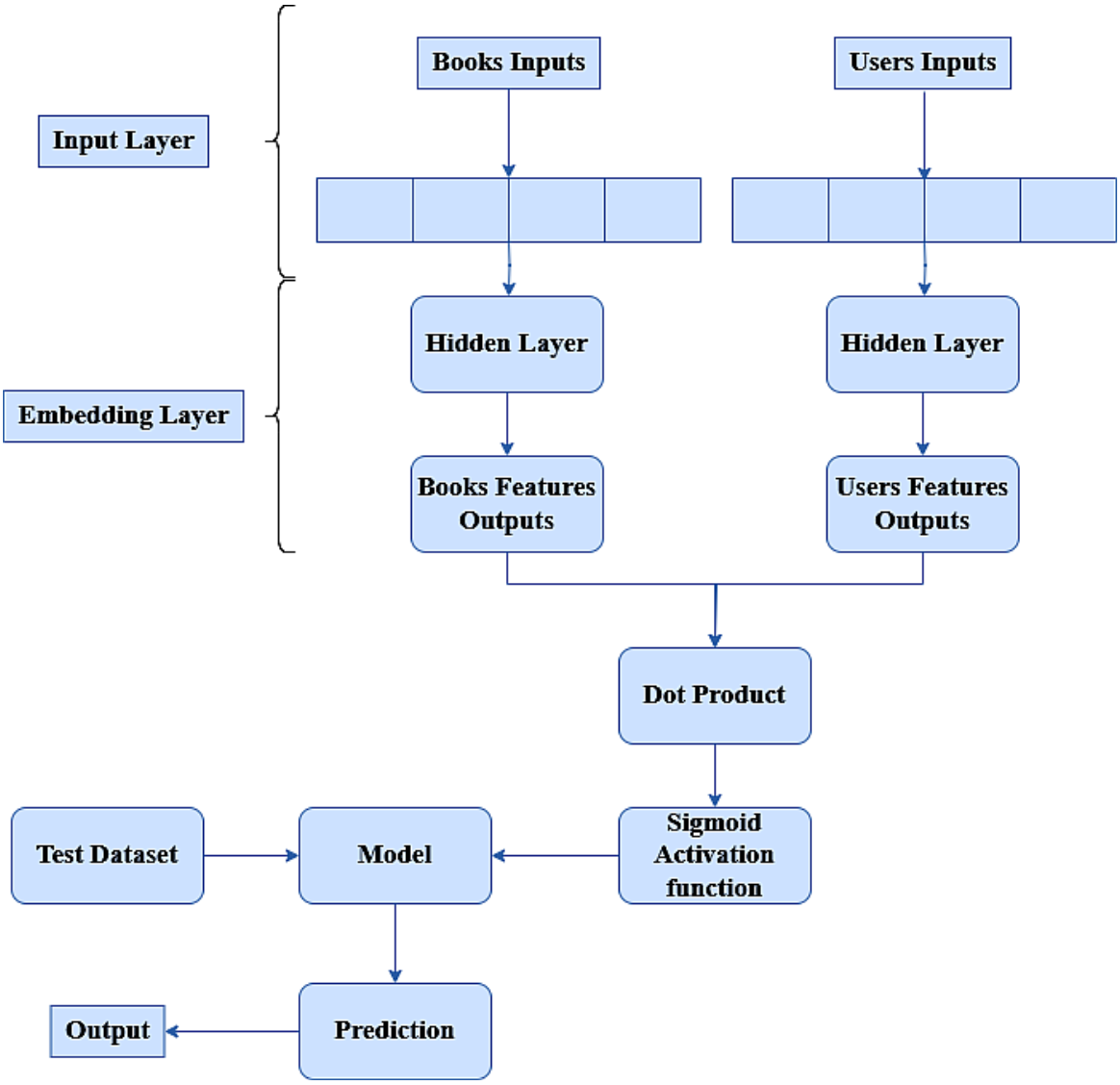

Figure 5: Model diagram

the parameters. This process allows the model to gradually reduce the error by making small, incremental changes to its weights. The updates are performed iteratively, typically for a predetermined number of epochs, or until convergence is achieved, meaning the model reaches a state where further updates no longer lead to significant improvements in performance.

In this section, we also provide an overview of the hyperparameters and optimization methods. Firstly, the choice of embedding size, set at 20 dimensions (`embedding_size=20`), reflects a balance between model complexity and the risk of overfitting, supported by empirical testing. The regularization technique used, L2 regularization (`tf.keras.regularizers.l2(1e-6)`), enhances generalization by penalizing large weights in the embedding matrices. Additionally, He Normal initialization (`he_normal`) ensures stable gradient propagation during training, which is crucial for model convergence. Optimization during training employs the Adam optimizer (`tf.keras.optimizers.Adam`) with a fixed learning rate of 0.001, chosen for its adaptive capabilities and efficiency in handling tasks like rating prediction. These choices are critical for ensuring reproducibility across experiments and provide insights into the model's design logic and optimization strategy, facilitating a deeper understanding of its performance and potential improvements.

5.2 Content-based technique

The content-based technique leverages user characteristics to generate recommendations. It suggests books by comparing them with the user profile. To identify similar books that align with the user's profile, we employed linear regression and logistic regression techniques. This approach effectively addresses the cold start problem by offering book recommendations to new users based on

Algorithm 1 Pseudo code of our proposed algorithm

Input:
Book crossing dataset
Settings:
Model Training Settings:
$batch\_size$: 32 {length of iteration}
$epochs$: 20
$verbose$: 1
Load and preprocess the dataset
Encode user and book IDs:
Create mappings for user and book IDs
Map the encoded IDs to the DataFrame
Normalize ratings:
Apply a lambda function to normalize ratings to a scale of 0 to 1
Split the data into training and validation sets:
Use train_test_split with an 80-20 split
Define the Recommender model:
Initialize embeddings for users and books
Initialize biases for users and books
Build the model's forward pass:
Compute user and book vectors and biases
Calculate the dot product of user and book vectors
Add biases and apply a sigmoid activation
Compile and train the model:
Compile the model with mean squared error loss and Adam optimizer
Train the model using the training data with specified batch size and epochs
Output:
Trained model

their location and age . In our linear and logistic regression models, we recorded results across various parameters. The Root Mean Square Error (RMSE) served as our evaluation metric, with the results presented in the tables below.

We wanted to select the best one and then used this model to recommend books to new users.

5.3 Methodology overview

This explanation outlines the key steps and settings used in training a recommender model, focusing on the hybridization step of our model based on the Book Crossing dataset.

Table 2: Linear regression results

Linear regression params	RMSE
n_jobs=2,positive=True	3.52
copy_X=True,positive=False	3.36
copy_X= False,positive=False	3.36
fit_intercept=True,positive=False1	3.36

Table 3: LOGISTIC regression results

Logistic regression params	RMSE
Without parameters	3.52
solver='newton-cg'	1.88
solver='liblinear'	1.88
solver='lbfgs'	1.87
solver='sag'	1.87
solver='saga'	1.87
penalty='l2'	1.87
penalty='none'	1.87
multiclass ='ovr'	1.88
multiclass ='multinomial'	1.87

5.3.1 Model training settings:

- **Batch Size:** Set to 32, indicating the number of samples processed before the model is updated.
- **Epochs:** Set to 20, representing the number of complete passes through the training dataset.
- **Verbose Level:** Set to 1, which provides detailed logging during training.

5.3.2 Hybrid recommendation function

In our hybrid model, we combine Collaborative Filtering (CF) and Content-Based (CB) methods by calculating individual recommendation scores for each approach. These scores are then combined using a weighted average to generate the final hybrid recommendation score. The formula for this hybrid score is:

$$\text{Hybrid Score} = \alpha \times \text{CF Score} + (1 - \alpha) \times \text{CB Score}$$

Here, α is a weighting factor that allows us to control the relative contribution of the CF and CB scores. By adjusting α, the model can be fine-tuned to optimize performance for different use cases, ensuring a balance between the two methods.

Pseudo code:

- Define function hybrid_score (CF, CB, α): `return` $\alpha \times \text{CF} + (1 - \alpha) \times \text{CB}$
- Define function get_recommendations (CF_scores, CB_scores, α): `final_scores = []`

  `For each item in CF_scores:`

```
        score = hybrid_score(CF_scores[item], CB_scores[item], α)
        final_scores.append((item, score))
    Sort final_scores by score in descending order
    return final_scores
```

6 Experimental results

The Root Mean Square Error (RMSE) [m] and Mean Absolute Error (MAE) [n]are the metrics used to analyze the results of the experiment. These objective measures are widely employed to evaluate the performance of recommendation system models. They are defined as follows:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(y_{\text{pred},i} - y_{\text{actual},i})^2}{n}}$$

The RMSE has the same measuring unit of the variable y. Mean Absolute Error (MAE). MAE is the average vertical distance between each point and the identity line. The formula is given below:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_{\text{pred},i} - y_{\text{actual},i}|$$

We further exemplify the loss equation in the subsequent manner:

$$Loss = \frac{\sum(y_{\text{pred}} - y_{\text{actual}})^2}{n}$$

In the above equations:

y_{pred} represents the model's predicted values, y_{actual} represents the dataset's actual values, $\sum$ denotes the sum of squared differences between predicted and actual values, and n represents the number of samples in the dataset.

Here's your text with corrections for grammar, clarity, and conciseness:

Multiple factors led us to select RMSE and MAE as the preferred metrics for RS. First, they are sensitive to prediction errors, allowing us to evaluate the predictive accuracy of our model regarding user preferences. Second, these metrics are easily interpretable because they are measured in the same units as the predicted and actual values, facilitating the communication of prediction errors to stakeholders and users. Third, MAE and RMSE possess desirable mathematical properties derived from the mean squared error, making them suitable for optimization objectives and providing a comprehensive evaluation of system performance [25][14]. Finally, these metrics demonstrate robustness to outliers, ensuring that extreme ratings or user preferences do not disproportionately influence the evaluation. By utilizing RMSE and MAE, we gain valuable insights into the performance of our recommender models.

To optimize the performance of our deep learning model, we conducted 20 training and validation iterations on a meticulously curated dataset. The purpose of these iterations was to enhance the model's ability to provide accurate recommendations by capturing complex data patterns and relationships.

At each epoch during the training process, we calculated the loss and mean squared error (MSE) for both the training and validation datasets. Our primary goal was to minimize the loss and MSE values, indicating improved accuracy and reliability in the model's suggestions.

Computational Cost:

We trained this model on Kaggle and assessed the training time. It took approximately 3 hours to yield results on our computer using a large dataset.

Memory and Computational Resource Usage: The computer used for training was equipped with an Intel Core i5-3320M processor (2.60 GHz, 2 cores, 4 threads) and 8 GB of DDR3 RAM. It is important to note that deep learning models typically demand significantly more computational resources and time compared to traditional methods.

Confidence Intervals and Standard Deviations:

- RMSE: The mean RMSE is 0.69, with a standard deviation of 0.031, and the 95% confidence interval is between 0.65 and 0.73. This means we are 95% confident that the true RMSE value lies within this range.
- MAE: The mean MAE is 0.51, with a standard deviation of 0.021, and the 95% confidence interval is between 0.49 and 0.53. Similarly, this indicates the range within which the true MAE value lies with 95% confidence. This demonstrates that the model's performance is reliable and we can expect similar results when the model is tested on new data.

The empirical evaluation of our RS demonstrates that it provides superior suggestions compared to existing works, achieving Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) values of 0.69 and 0.51, respectively. The graph above illustrates these findings by depicting the progression of the mean squared error across the epochs. It's important to note that a lower RMSE value indicates better forecast accuracy for the target variable. These results provide compelling evidence of our deep learning model's ability to reliably predict the desired variable.

The following graphical representation of the loss functions for the training and validation sets offers compelling evidence of the network's efficient training.

7 Results discussion

In this section, we detailed comparison between our results and the SOTA ones presented earlier in Section 2. For the

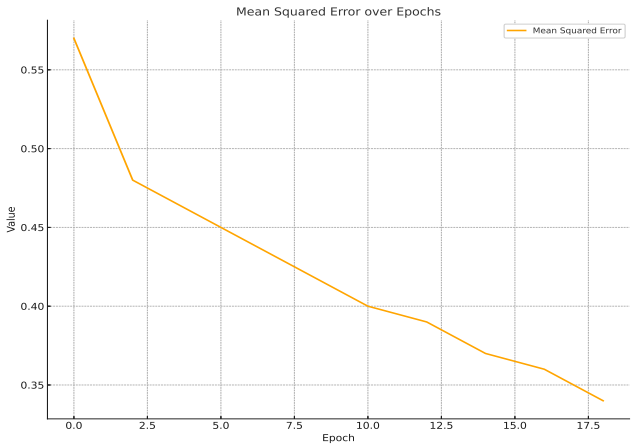

Figure 6: Mean squared error by epoch

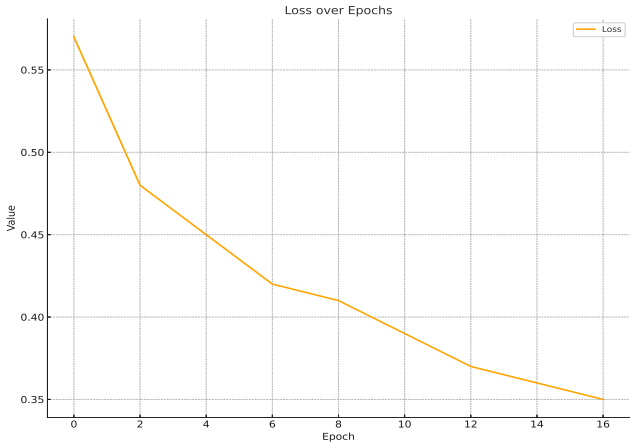

Figure 7: Training and testing losses in 20 epochs

techniques that use the same evaluation metrics as our work and the same Book Crossing dataset, Our model shows good results with RMSE and MAE values of 0.69 and 0.51, respectively, as shown in Table 4, outperforming other techniques (e.g., [[2], [7], [26]]).

The experimental results demonstrate that embedding layers improves the performance of our model. The table below illustrates the Comparative Results section, followed by an illustrative diagram.

Table 4: Comparison results

Method	**RMSE**	**MAE**
Auto-Surprise (TPE) [2]	3.52	2.88
Auto-Surprise (ATPE)[2]	3.51	2.87
Regular Matrix Factorization (MF) [7]	0.70	0.45
k-NN prediction model[26]	2.99	2.63
Our Model	0.69	0.51

Finally, concerning scalability, our method achieves better results due to:

- Embedding Layers:
  These layers transform categorical data (like user IDs and book IDs) into dense vectors, enabling the model to capture complex relationships in a continuous space. This reduces the sparsity of user-item interactions and enhances learning of detailed user preferences, leading to significantly lower RMSE and MAE compared to traditional methods.

- Bias Terms: Incorporating bias terms for users and books allows the model to adjust predictions based on inherent preferences and popularity, improving accuracy by better aligning predicted ratings with actual user interactions.

- Combination of CF and CB Techniques: By integrating both approaches, the model gains a comprehensive understanding of user preferences and item attributes. This synergistic approach enhances personalized recommendations, resulting in reduced prediction errors compared to single-method models.

- Deep Learning Framework: Utilizing deep learning facilitates effective learning from large datasets, enabling the model to capture intricate patterns and relationships that simpler algorithms might overlook. This capability enhances generalization and improves the reliability of recommendations.

The hybrid recommendation model proposed surpasses existing methods through several key architectural strengths:

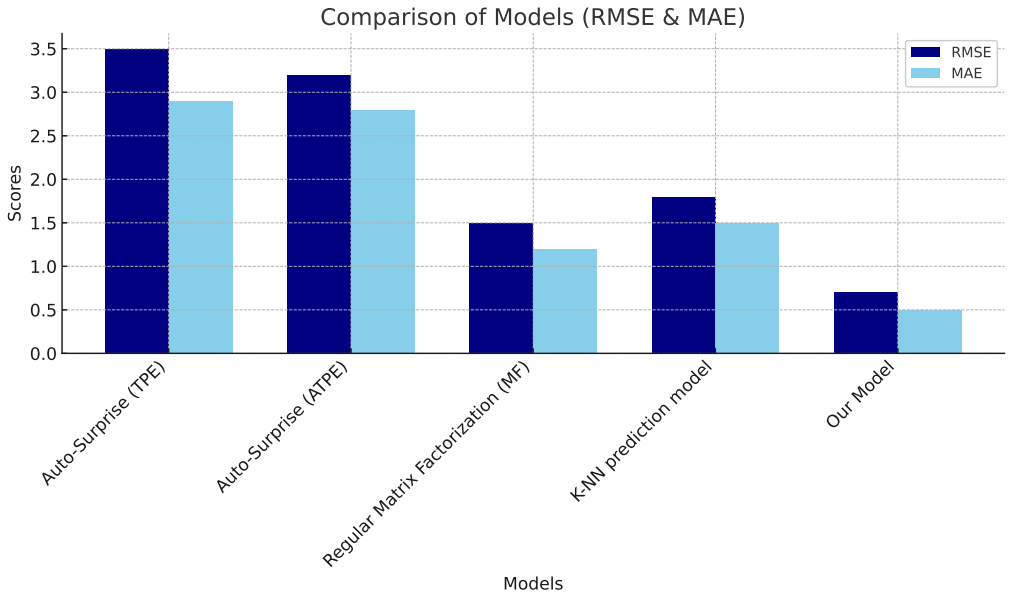

Figure 8: Comparative results graph

Together, these features contribute to superior performance, as demonstrated by significantly reduced RMSE (0.69) and MAE (0.51) values. They highlight the model's efficacy in delivering precise, personalized, and context-aware book recommendations.

8 Conclusion and future directions

The current era is characterized by the widespread availability of online educational resources, leading to the significant challenge of obtaining relevant information for individuals engaged in e-learning. The convergence of adaptive e-learning and personalized technology has fostered the emergence of innovative solutions, with RS becoming a powerful tool. These systems are designed to meet the needs of learners in ever-evolving environments while alleviating the problem of information overload.

Our comprehensive investigation delved into the domain of e-book RS, a crucial aspect of online education. The objective of our endeavor was to develop a hybrid e-book RS that seamlessly integrates CB machine learning with the depth of deep learning embedding layers. This initial phase of our journey focuses on enhancing the reading experience through CB book recommendations tailored to readers' preferences.

Encouragingly, our rigorous testing, which included cold-start scenarios and data sparsity, has yielded promising results, reinforcing the system's efficacy in the realm of e-books. The empirical analysis of our RS on a sizable e-book dataset demonstrates that it outperforms comparable existing works, achieving a Mean Absolute Error (MAE) of 0.51 and a Root Mean Square Error (RMSE) of 0.69. In our forthcoming efforts, we pledge to:

- Augment the system's intelligence by deepening the understanding of learner behaviors, thereby suggesting books that align with their interests.
- Expand the scope of content-based recommendations by considering a broader range of characteristics, including the learner's intended study direction and intellectual level.
- Continuously improve the system's performance by exploring alternative methodologies and integrating advanced deep learning approaches.
- Evolve the model's architecture and layer configuration to further enhance the system's capabilities and its ability to deliver superior results.

In summary, our exploration of e-book recommendation systems reflects our dedication to enriching the educational experience. By overcoming challenges, embracing innovation, and prioritizing user-centricity, we envision a future where our recommendations shape learning journeys and cultivate a landscape of personalized growth.

9 Acknowledgement

The authors extend their heartfelt gratitude to the Laboratory of Computer Science and Mathematics at the University of Souk Ahras for their invaluable support. We also wish to express our deep appreciation to all esteemed colleagues and professors who generously contributed their expertise, guidance, and assistance. Their contributions have profoundly enriched our understanding of recommendation systems and significantly enhanced the quality of this paper's revision.

References

[1] Zafar Ali, Shah Khusro, and Irfan Ullah. A hybrid book recommender system based on table of contents (toc) and association rule mining. In *Proceedings of the 10th International Conference on Informatics and Systems*, pages 68–74, 2016. doi:10.1145/2908446.2908481.

[2] Rohan Anand and Joeran Beel. Auto-surprise: An automated recommender-system (autorecsys) library with tree of parzens estimator (tpe) optimization. In *Proceedings of the 14th ACM Conference on Recommender Systems*, pages 585–587, 2020. doi:10.1145/3383313.3411467.

[3] Oren Barkan and Noam Koenigstein. Item2vec: neural item embedding for collaborative filtering. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2016. doi:10.1109/mlsp.2016.7738886.

[4] Peter Brusilovski, Alfred Kobsa, and Wolfgang Nejdl. *The adaptive web: methods and strategies of web personalization*, volume 4321. Springer Science & Business Media, 2007. doi:10.1007/978-3-540-72079-9.

[5] Robin Burke. Hybrid web recommender systems. *The adaptive web: methods and strategies of web personalization*, pages 377–408, 2007. doi:10.1007/978-3-540-72079-9_12.

[6] Andrei Ionut DAMIAN, Laurentiu Gheorghe PICIU, Nicolae TAPUS, and Bogdan DUMITRESCU. Deep recommender engine based on efficient product embeddings neural pipeline. 2019. doi:10.1109/roedunet.2018.8514141.

[7] Khishigsuren Davagdorj, Kwang Ho Park, and Keun Ho Ryu. A collaborative filtering recommendation system for rating prediction. In *Advances in Intelligent Information Hiding and Multimedia Signal Processing: Proceedings of the 15th International Conference on IIH-MSP in conjunction with the 12th International Conference on FITAT, July 18-20, Jilin, China, Volume 1*, pages 265–271. Springer, 2019. doi:10.1007/978-981-13-9714-1_29.

[8] Surabhi Dwivedi and VS Kumari Roshni. Recommender system for big data in education. In *2017 5th National Conference on E-Learning & E-Learning Technologies (ELELTECH)*, pages 1–4. IEEE, 2017. doi:10.1109/eleltech.2017.8074993.

[9] Ashkan Ebadi and Adam Krzyzak. A hybrid multi-criteria hotel recommender system using explicit and implicit feedbacks. *International Journal of Computer and Information Engineering*, 10(8):1450–1458, 2016. doi:10.1145/3456146.3456157.

[10] Aurora Esteban, Amelia Zafra, and Cristóbal Romero. Helping university students to choose elective courses by using a hybrid multi-criteria recommendation system with genetic optimization. *Knowledge-Based Systems*, 194:105385, 2020. doi:10.1016/j.knosys.2019.105385.

[11] Matjaž Gams and Tine Kolenik. Relations between electronics, artificial intelligence and information society through information society rules. *Electronics*, 10(4):514, 2021. doi:10.3390/electronics10040514.

[12] Pijitra Jomsri. Book recommendation system for digital library based on user profiles by using association rule. In *Fourth edition of the International Conference on the Innovative Computing Technology (INTECH 2014)*, pages 130–134. IEEE, 2014. doi:10.1109/intech.2014.6927766.

[13] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 79–86, 2010. doi:10.1145/1864708.1864727.

[14] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009. doi:10.1109/mc.2009.263.

[15] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003. doi:10.1109/mic.2003.1167344.

[16] Faiz Maazouzi, Hafed Zarzour, and Yaser Jararweh. An effective recommender system based on clustering technique for ted talks. *International Journal of Information Technology and Web Engineering (IJITWE)*, 15(1):35–51, 2020. doi:10.4018/ijitwe.2020010103.

[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013. doi:10.7551/mitpress/1120.003.0018.

[18] ThaiBinh Nguyen and Atsuhiro Takasu. Npe: neural personalized embedding for collaborative filtering. *arXiv preprint arXiv:1805.06563*, 2018. doi:10.24963/ijcai.2018/219.

[19] Bo Ni and Xiaona Xie. Distributed distribution and scheduling of teaching resources based on a random matrix educational leadership model. *Informatica*, 48(8), 2024. doi:10.31449/inf.v48i8.5440.

[20] Carlos Porcel, Alberto Ching-Lopez, Juan Bernabe-Moreno, Alvaro Tejeda-Lorente, and Enrique Herrera-Viedma. Fuzzy linguistic recommender systems for the selective diffusion of information in digital libraries. *Journal of Information Processing Systems*, 13(4), 2017. doi:10.3745/jips.04.0035.

[21] Xiaosi Qi, Jianwei Zhao, and Guochao Hu. Explore the personalized resource recommendation of educational learning platforms: Deep learning. *Informatica*, 48(7), 2024. doi:10.31449/inf.v48i7.5690.

[22] Ahmed H Ragab and Passant El-Kafrawy. Embedding based recommender systems, a review and comparison. *The Egyptian Journal of Language Engineering*, 9(1):1–11, 2022. doi:10.21608/ejle.2022.91884.1025.

[23] Sushama Rajpurkar, Darshana Bhatt, Pooja Malhotra, MSS Rajpurkar, and MDR Bhatt. Book recommendation system. *International Journal for Innovative Research in Science & Technology*, 1(11):314–316, 2015. doi:10.38124/ijisrt/ijisrt24sep118.

[24] Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer, 2010. doi:10.1007/978-0-387-85820-3_1.

[25] Francesco Ricci, Lior Rokach, and Bracha Shapira. Recommender systems: introduction and challenges. *Recommender systems handbook*, pages 1–34, 2015. doi:10.1007/978-1-4899-7637-6_1.

[26] Rohit, Sai Sabitha, and Tanupriya Choudhury. Proposed approach for book recommendation based on user k-nn. In *Advances in Computer and Computational Sciences: Proceedings of ICCCCS 2016, Volume 2*, pages 543–558. Springer, 2018. doi:10.1007/978-981-10-3773-3_53.

[27] John K Tarus, Zhendong Niu, and Dorothy Kalui. A hybrid recommender system for e-learning based on context awareness and sequential pattern mining. *Soft Computing*, 22:2449–2461, 2018. doi:10.1007/s00500-017-2720-6.

[28] Aleš Tavčar, Antonya Csaba, and Eugen Valentin Butila. Recommender system for virtual assistant supported museum tours. *Informatica*, 40(3), 2016. doi:10.58680/tetyc201323610.

[29] Donghui Wang, Yanchun Liang, Dong Xu, Xiaoyue Feng, and Renchu Guan. A content-based recommender system for computer science publications. *Knowledge-Based Systems*, 157:1–9, 2018. doi:10.1016/j.knosys.2018.05.001.

[30] Xiuhui Wang. Personalized recommendation system of e-learning resources based on bayesian classification algorithm. *Informatica*, 47(3), 2023. doi:10.31449/inf.v47i3.3979.

[31] Hafed Zarzour, Sabrina Bendjaballah, and Hadjer Harirche. Exploring the behavioral patterns of students learning with a facebook-based e-book approach. *Computers & Education*, 156:103957, 2020. doi:10.1016/j.compedu.2020.103957.

[32] Hafed Zarzour, Faiz Maazouzi, Mohammad Al-Zinati, Amjad Nusayr, Mohammad Alsmirat, Mahmoud Al-Ayyoub, and Yaser Jararweh. Using k-means clustering ensemble to improve the performance in recommender systems. In *2022 International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*, pages 176–180. IEEE, 2022. doi:10.1109/idsta55301.2022.9923070.

[33] Hafed Zarzour, Faiz Maazouzi, Mohamed Soltani, and Chaouki Chemam. An improved collaborative filtering recommendation algorithm for big data. In *Computational Intelligence and Its Applications: 6th IFIP TC 5 International Conference, CIIA 2018, Oran, Algeria, May 8-10, 2018, Proceedings 6*, pages 660–668. Springer, 2018. doi:10.1007/978-3-319-89743-1_56.

[34] Hao Zhang, Yingyuan Xiao, and Zhongjing Bu. Personalized book recommender system based on chinese library classification. In *2017 14th Web Information Systems and Applications Conference (WISA)*, pages 127–131. IEEE, 2017. doi:10.1109/wisa.2017.42.

[35] Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*, pages 22–32, 2005. doi:10.1145/1060745.1060754.