Comparison of Hybrid Book Recommender Systems: Matrix Factorization with Neural Networks vs. Neural Collaborative Filtering with Attention

Alisa Todorova
Department of Computer Science
Faculty of Science, Technology and Medicine
University of Luxembourg

***Abstract*—In this paper, we explore two advanced hybrid models for book recommendation systems: Matrix Factorization with Neural Networks (MF-NN) and Neural Collaborative Filtering with Attention (NCF-Attention). The goal of this research is to enhance prediction accuracy and generalization by leveraging deep learning techniques alongside traditional collaborative filtering methods. Our study employs the "Book-Crossing: User review ratings" dataset from Kaggle to train, optimize, incrementally enhance, and then evaluate each of the models, focusing on their performance and scalability. Our findings indicate that while both models offer significant improvements over traditional approaches, the MF-NN model demonstrates superior performance in terms of accuracy and computational efficiency for our given dataset.**

***Index Terms*—Recommender Systems, Collaborative Filtering, Matrix Factorization, Neural Networks, Neural Collaborative Filtering, Attention Mechanism, Deep Learning, Hybrid Recommender Systems, Book Recommendation**

I. Introduction

Reading books has regained popularity as a hobby thanks to social media. Hashtags like "#BookTok" on TikTok and "#Bookstagram" on Instagram have influenced teenagers and young adults to purchase and read more books [7] [14]. With millions of book recommendations available, finding one you will enjoy amongst the information overload can be challenging. The solution lies in book recommender systems, which provide personalized suggestions based on a user's past preferences and behaviors, such as purchasing, rating, or reviewing books. Famous e-commerce platforms using these systems include Goodreads and Amazon. Accurate recommendations can help users discover new books that align with their tastes, thus, increasing user satisfaction and engagement. Furthermore, the success of recommendation algorithms commercially impacts digital bookstores by increasing sales and keeping customers.

Collaborative filtering (CF) is one of the most popular techniques in recommender systems. It generates recommendations based on the user's past interactions with items, while disregarding user and item content information [8]. Amongst all CF techniques, Matrix factorization (MF) is the most commonly used due to its scalability, flexibility, and ease of implementation. MF learns a latent space - an abstract representation that captures underlying patterns in user-item interactions, allowing the recommender system to represent users and items and predict personalized rankings for each user based on their similarities [11]. Despite its effectiveness, the simple interaction function in MF is often inadequate for modeling the complex relationships between users and items [8]. That is why, hybrid approaches incorporating neural networks (NN) have been proposed to address the limitations of individual algorithms. Neural networks (NN) are a class of machine learning algorithms capable of learning complex non-linear relationships between inputs and outputs. When employed in recommender systems, they enhance prediction accuracy by modeling the non-linear relationships between user-item features and capturing their dependencies [12]. State-of-the-art hybrid methods, such as NeuMF, eALS, ANNCF, SLAN, excel at learning latent factors, especially when there is a large volume of ratings and rich supplementary information [4] [11].

This paper explores two different hybrid models: Matrix Factorization with Neural Networks (MF-NN) and Neural Collaborative Filtering with Attention (NCF-Attention). MF-NN leverages collaborative filtering with deep learning for user-item interaction modeling. NCF-Attention leverages a deep learning model for collaborative filtering while incorporating an attention mechanism to focus on more relevant user-item interactions. An attention mechanism allows the neural network to focus on the most relevant interactions, thus, prioritizing different elements during computation.

In real-world scenarios, achieving high accuracy and generalization in recommendation systems still remains challenging. Often models have issues with scalability, high computational complexity, and "cold start" problems (i.e., the challenge of making accurate recommendations when there is insufficient data available). The goal of this research is to compare and determine the most suitable approach between the two

This report was prepared for the course *Introduction to Deep Learning*, which is part of the Master in Information and Computer Sciences. Supervisors: Luis A. Leiva, Bereket Abera Yilma.

proposed models for the dataset "Book-Crossing: User review ratings" from Kaggle [2]. This is accomplished through extensive training, optimization, and incremental enhancing of each model to enhance performance and identify the superior one. Hence, this research contributes to the field of recommender systems by providing a detailed comparison of two advanced deep learning-based models: MF-NN and NCF-Attention. By evaluating these models on a large dataset of real book ratings, we offer insights into their performance, scalability, and suitability for real-world applications.

This paper is structured as follows: Section II reviews related works. Section III describes our mythology, including data preparation, model evaluation, and optimization techniques. Section IV describes the conducted experiments. In section V, the results from the experiments are presented, which are then interpreted and explained in section VI. Section VII discusses potential limitations in our research and suggests directions for future research. Finally, section VIII gives a summary of this research.

II. Related Works

Xue et al. [11] proposed a novel matrix factorization model with a neural network architecture for improving top-N recommendations. Their model outperformed state-of-the-art methods in several benchmark datasets by effectively utilizing both explicit ratings and implicit feedback through a new loss function based on binary cross-entropy. However, their model was too complex, which made it computationally expensive for very large datasets. Furthermore, the model could exhibit potential overfitting due to the depth of the neural network layers. The model we propose in this paper, Matrix Factorization with Neural Networks (MF-NN), implements a robust dataset split strategy that ensures thorough evaluation and prevents overfitting. Additionally, the use of L2 regularization and early stopping during model training further mitigate overfitting. Furthermore, we perform experiments with different embedding dimensions and training callbacks to optimize model performance without significantly increasing computational complexity.

Yadav et al. [12] proposed a novel hybrid recommendation system that combines matrix factorization with neural networks to improve the accuracy and coverage of recommendations. Their model outperformed individual MF and NN models, as well as other state-of-the-art algorithms in terms of Root Mean Squared Error and coverage on multiple datasets. However, their model needs large datasets to train the neural networks effectively, which proved to be computationally expensive, especially with increasing data size and complexity. The model we propose in this paper, Matrix Factorization with Neural Networks (MF-NN), implements a robust data split strategy that ensures sufficient data for training while avoiding overfitting. Additionally, the use of early stopping, L2 regularization, and dynamic learning rate adjustments helps with managing computational complexity and improving the model's generalization capabilities, ensuring effective performance even with smaller datasets.

Nassar et al. [8] proposed a multi-criteria collaborative filtering recommender system by fusing deep neural networks and matrix factorization to enhance recommendation accuracy and coverage. Their model outperformed state-of-the-art methods in terms of improvement in recommendation accuracy and coverage. However, their models needs large amounts of data to effectively train the deep learning components and the computational complexity of the fused model. This could pose challenges for scalability and real-time applications. The model we propose in this paper, Matrix Factorization with Neural Networks (MF-NN), implements a robust dataset split strategy to ensure comprehensive model evaluation and prevent overfitting. The use of early stopping, L2 regularization, and dynamic learning rate adjustments optimizes model training and improves generalization, making the model more efficient and scalable even with smaller datasets.

Dong et al. [4] proposed a hybrid collaborative filtering model that integrates deep learning with matrix factorization to effectively utilize side information and improve recommendation performance. They developed an additional stacked denoising autoencoder (aSDAE) that enhances latent factor extraction from both side information and the rating matrix, significantly outperforming other state-of-the-art methods in recommendation accuracy and recall. However, their models needs large amounts of data to effectively train the deep learning components and the computational complexity of the fused model. This could pose challenges for scalability and real-time applications. The model we propose in this paper, Matrix Factorization with Neural Networks (MF-NN), implements a robust dataset split strategy to ensure comprehensive model evaluation and prevent overfitting. The use of early stopping, L2 regularization, and dynamic learning rate adjustments optimizes model training and improves generalization, making the model more efficient and scalable even with smaller datasets.

Aljunid and Dh [1] proposed an efficient deep learning approach for collaborative filtering recommender systems, focusing on improving performance by addressing matrix sparsity and scalability issues. Their deep learning-based collaborative recommender system (DLCRS) achieved better performance in terms of Root Mean Square Error (RMSE) compared to existing methods, demonstrating the effectiveness of deep learning in recommendation systems without requiring extra information beyond user-item interactions. However, their model shows potential overfitting due to the complexity of deep learning models and the requirement for significant computational resources for training, particularly as the number of epochs and data size increase. The model we propose in this paper, Matrix Factorization with Neural Networks (MF-NN), implements a robust dataset split strategy to ensure comprehensive model evaluation and prevent overfitting. The use of early stopping, L2 regularization, and dynamic learning rate adjustments optimizes model training and improves generalization, making the model more efficient and scalable even with smaller datasets.

Xia et al. [10] proposed a neural collaborative filtering recommender method using a Stacked Denoising Auto Encoder

(SDAE) and Gated Recurrent Unit (GRU) enhanced with an attention mechanism to improve recommendation performance. Their model effectively integrates user-item rating information and auxiliary information, outperforming traditional and existing deep learning-based recommendation methods. However, due to the integration of multiple deep learning components and auxiliary information, their model has increased complexity and computational costs, and is not scalable for very large datasets or real-time applications. Additionally, the model's effectiveness heavily relies on the quality and availability of auxiliary information, which might not always be present. The model we propose in this paper, Neural Collaborative Filtering with Attention (NCF-Attention), systematically explores and evaluates different architectural choices, hyperparameters, and regularization techniques to enhance model robustness and scalability. By leveraging L2 regularization, dropout, and learning rate adjustments, we manage model complexity and improve generalization, ensuring that our model performs well even with varying quality and availability of auxiliary information.

Yang et al. [13] proposed a novel hybrid recommendation model called GANCF, which integrates deep learning techniques with traditional collaborative filtering to enhance recommendation performance. Their model combines user and item auxiliary information with deep learning methods such as attention mechanisms and GRU, and significantly outperforms traditional recommendation models. However, their model cannot account for temporal dynamics of user preferences. It also has limited integration of multi-source heterogeneous data, which can affect the model's ability to adapt to changes in user behavior over time. The model we propose in this paper, Neural Collaborative Filtering with Attention (NCF-Attention), incorporates a time-aware prediction mechanism to capture the evolving nature of user preferences.

Chen et al. [3] proposed an Attribute-based Neural Collaborative Filtering (ANCF) method that integrates user and item attribute information using an attention mechanism to improve recommendation accuracy by capturing high-order nonlinear relationships between users and items. Their model shows significantly improvement in the performance of recommendation systems on multiple datasets compared to traditional methods. However, their model has increased computational complexity due to the incorporation of attribute information and the attention mechanism, which may lead to longer training times and higher resource consumption. Additionally, their model may struggle with "cold-start" problems if attribute information is sparse or unavailable for new users or items. The model we propose in this paper, Neural Collaborative Filtering with Attention (NCF-Attention), systematically explores different architectural choices, hyperparameters, and regularization techniques to optimize the balance between model complexity and performance. By incrementally enhancing the models and incorporating dropout and regularization, we aim to improve generalization and reduce overfitting, thereby making the model more robust even with sparse data.

Wang et al. [9] proposed a time-aware deep collaborative filtering framework that models dynamic user preferences using an attention mechanism and predicts matching scores using deep learning models. Their model dynamically models short-term and long-term user preferences, significantly improving the accuracy of top-k recommendations. However, there is significant computational complexity introduced by the deep learning models. Furthermore, their temporal sequence of interactions might not always capture the full scope of user preferences due to potential noise in interaction timestamps. The model we propose in this paper, Neural Collaborative Filtering with Attention (NCF-Attention), systematically explores and evaluates different architectural choices, hyperparameters, and regularization techniques across three models. Each model iteration enhances the previous one, incorporating dropout and various regularization strategies to improve robustness and reduce overfitting. Additionally, our model implements attention mechanisms to refine focus on the most relevant parts of the interaction data, further improving prediction accuracy and generalization to unseen data.

Lv et al. [6] proposed an attention-based item collaborative filtering model (AICF) that uses three different attention mechanisms to estimate the weights of historical items that users have interacted with, aiming to improve the accuracy of recommendation systems. Their model, especially with the Self-Attention mechanism, significantly improves recommendation accuracy over state-of-the-art models on both dense and sparse datasets. However, their model shows increased computational complexity and training time due to the multi-layer attention mechanisms and the necessity of pre-training to achieve optimal performance, which may not be feasible for all datasets. The model we propose in this paper, Neural Collaborative Filtering with Attention (NCF-Attention), systematically explores and evaluates different architectural choices, hyperparameters, and regularization techniques in the NCF-Attention models. Specifically, we incorporate deeper layers with optimized regularization and use dropout techniques to reduce overfitting and computational complexity, aiming to achieve better generalization and predictive accuracy without relying heavily on pre-training.

III. Methods

In this section, we outline the process of data acquisition and preparation, including data cleaning, formatting, and splitting into training, validation, and test sets. We then describe our two models' architectures and the techniques used for evaluation and optimization, such as regularization, early stopping, and learning rate adjustments. Additionally, we provide an overview of the metrics used to assess model performance. In the end, the process of recommending books to a user is explained.

A. Data Preparation

1) Dataset: The dataset used in this research is "Book-Crossing: User review ratings" from Kaggle [2]. It is made up of three main datasets:

- **Users:** consists of 278,858 entries, each representing a unique user ID, location (city, state, and country), and age.
- **Books:** consists of 271,379 entries, detailing each book's ISBN, title, author, year of publication, publisher, and URLs for small, medium, and large images.
- **Ratings:** consists of 1,149,780 entries, where each entry records a user ID, the book's ISBN, and the corresponding user's rating, which is a value from 0 to 10.

Some interesting findings from these datasets are:

- 50.1% of the users are from the USA.
- The average age is 35 years old.
- The top 5 most rated authors are Stephen King with 7664 ratings, Nora Roberts with 6407 ratings, John Grisham with 4168 ratings, James Patterson with 4096 ratings, and Mary Higgins Clark with 3372 ratings.
- The top 5 most rated books are "Wild Animus" by Rich Shapero with 2141 ratings, "The Lovely Bones: A Novel" by Alice Sebold with 920 ratings, "The Da Vinci Code" by Dan Brown with 708 ratings, "Bridget Jones's Diary" by Helen Fielding with 599 ratings, and "The Nanny Diaries: A Novel" by Emma McLaughlin and Nicola Kraus with 573 ratings.

These datasets are interconnected through the **user_id** and **isbn** columns. To facilitate the analysis of user preferences and book ratings, these datasets were merged sequentially. Initially, the *users* and *ratings* datasets were merged on the **user_id** column. Then, the resulting DataFrame was merged with the *books* dataset on the **isbn** column, creating a unified DataFrame that integrates user demographics, book details, and user ratings.

2) Pre-processing of the data: To prepare the dataset for analysis and ensure consistency in data types, several preprocessing steps were undertaken.

Initially, we checked for missing values in the newly merged DataFrame. After inspection, we decided to drop all missing values, as most were found in the **age** column. Given that our models and analyses are focused on the users' book preferences based on their book ratings, we determined that the absence of age values for some users would not significantly impact our study.

Furthermore, the data types of key columns were converted: **user_id** was converted to an integer type to ensure uniformity; **isbn** was converted to a string type to preserve the integrity of book identifiers; **rating** was converted to a float type to accommodate potential decimal values; and **year_of_publication** was converted to an integer type for consistency in temporal analysis. Following the type conversions, **user_id** and **isbn** were re-indexed to facilitate efficient processing and analysis. Unique mappings were created for both **user_id** and **isbn**, assigning each unique user and book a sequential integer identifier. This was achieved by creating dictionaries that mapped the original **user_id** and **isbn** values to new indices using enumeration. The DataFrame was then updated to reflect these new mappings, effectively transforming **user_id** and **isbn** into re-indexed integers.

B. Model Evaluation and Optimization Techniques

*1) **Regularization techniques**:*

- L1 Regularization (Lasso) adds a penalty based on the absolute value of coefficients. This encourages sparsity.
- L2 Regularization (Ridge) adds a penalty based on the square of the coefficients. This encourages smaller and more evenly distributed weights.
- Early dropout: Early on during training, typically closer to the input layer, randomly selected neurons are ignored.
- Early stopping: The training process is halted when the values of the validation dataset start to degrade.

*2) **Optimizer**:* Adam optimizer is an adaptive learning rate optimization algorithm for training neural networks. It helps the model to efficiently converge on optimal parameter values [5].

*3) **Evaluation metrics**:* In the following formulas we have: n is the number of observations, Y is the vector of observed values of the variable being predicted, and $\hat{Y}$ represents the predicted values.

- Mean Squared Error: $MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2$
- Test and validation RMSE: $\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2}$

*4) **Activation functions** [11]:*

- *ReLU*: $f(x) = max(0, x)$.
- *Leaky ReLU*: $f(x) = \begin{cases} x & \text{if } x > 0, \\ 0.01x & \text{if otherwise.} \end{cases}$

*5) **Learning rate adjustment**:* *ReduceLROnPlateau* reduces the learning rate when a metric has stopped improving after a certain number of allowed epochs with no improvement, also known as "patience".

C. Models

1) Parameters: To ensure robust model evaluation and prevent underfitting and overfitting, the dataset was split into training, validation, and test sets.

First, the original dataset was divided into two parts: 80% for the training set and 20% for the test set. This initial split ensures that a substantial portion of the data was reserved for evaluating the final model's performance. The *train_test_split* function from the *sklearn* library was used for this purpose, with a *random_state* parameter set to 42 to ensure reproducibility of the splits.

Next, the training set was further split into two parts: 80% for the training set and 20% for the validation set. The validation set was used during the model training process to tune hyperparameters and make decisions about model adjustments without using the test set. This split was also performed using the *train_test_split* function, again with a *random_state* parameter set to 42 to maintain consistency.

In the end, we had three datasets:

- The training set, which comprises 64% of the original dataset.
- The validation set, which comprises 16% of the original dataset.
- The test set, which comprises 20% of the original dataset.

With the data split, the next step involved extracting the input arrays for the training and validation sets. This extraction process involves selecting the columns **user_id**, **isbn**, and **rating** from the training and validation DataFrames and converting them to NumPy arrays. These arrays were used as inputs for the machine learning model.

Additionally, the total number of unique users and items was determined from the mappings created earlier. This information was used for defining the dimensions of the embedding layers in our models. The embedding dimension is a hyperparameter that defined the size of the vectors into which user and item IDs would be mapped. The choice of embedding dimension can significantly impact the model's performance. During our experiments (as seen in section IV), we tested different embedding dimensions to identify the optimal value through rigorous experimentation and validation.

2) Matrix Factorization with Neural Networks (MF-NN): We created a Matrix Factorization with Neural Networks (MF-NN) model to leverage collaborative filtering combined with deep learning for user-item interaction modeling. This approach captures the latent features of users and items (i.e., books) through embeddings. It then integrates these features using a deep neural network to predict user ratings for items. The model architecture is detailed as follows:

- **User and Item Embeddings:** The model first took user and item IDs as input, each represented as a single integer. These IDs were then mapped to dense vectors of a specified dimension (i.e., **embedding_dim**) using embedding layers. L2 regularization with a coefficient of 0.01 was applied to the embedding layers to penalize large weights and reduce overfitting.
- **Flattening:** The embedding vectors for users and items were flattened to create single-dimensional vectors.
- **Concatenation:** The newly flattened user and item vectors were concatenated to form a single combined vector.
- **Dense Layers:** This concatenated vector was passed through two dense, fully connected layers with 128 and 64 neurons, respectively. Each dense layer included a *ReLU* activation function and was regularized using L2 regularization with a coefficient of 0.01 to further mitigate overfitting.
- **Output Layer:** The final output layer consisted of a single neuron without an activation function, which predicted the rating as a continuous value. It also included L2 regularization with a coefficient of 0.01.

The model was compiled with the Adam optimizer and Mean Squared Error (MSE) loss function to train on the rating prediction task. Tables I and II provide a detailed summary of the MF-NN model's architecture with the two different embedding layers, respectively, tested in this paper.

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 1)	0	-
input_layer_1 (InputLayer)	(None, 1)	0	-
embedding (Embedding)	(None, 1, 50)	2,694,400	input_layer[0][0]
embedding_1 (Embedding)	(None, 1, 50)	11,450,650	input_layer_1[0][0]
flatten (Flatten)	(None, 50)	0	embedding[0][0]
flatten_1 (Flatten)	(None, 50)	0	embedding_1[0][0]
concatenate (Concatenate)	(None, 100)	0	flatten[0][0], flatten_1[0][0]
dense (Dense)	(None, 128)	12,928	concatenate[0][0]
dense_1 (Dense)	(None, 64)	8,256	dense[0][0]
dense_2 (Dense)	(None, 1)	65	dense_1[0][0]

TABLE I: A detailed summary of the MF-NN model's architecture with 50 embedding layers

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 1)	0	-
input_layer_1 (InputLayer)	(None, 1)	0	-
embedding (Embedding)	(None, 1, 20)	1,077,760	input_layer[0][0]
embedding_1 (Embedding)	(None, 1, 20)	4,580,260	input_layer_1[0][0]
flatten (Flatten)	(None, 20)	0	embedding[0][0]
flatten_1 (Flatten)	(None, 20)	0	embedding_1[0][0]
concatenate (Concatenate)	(None, 40)	0	flatten[0][0], flatten_1[0][0]
dense (Dense)	(None, 128)	5,248	concatenate[0][0]
dense_1 (Dense)	(None, 64)	8,256	dense[0][0]
dense_2 (Dense)	(None, 1)	65	dense_1[0][0]

TABLE II: A detailed summary of the MF-NN model's architecture with 20 embedding layers

3) Neural Collaborative Filtering with Attention (NCF-Attention): The Neural Collaborative Filtering with Attention (NCF-Attention) model enhances the predictive performance for user-item interactions. We created three NCF-Attention models with incremental improvements in order to perform systematic exploration and evaluation of different architectural choices, hyperparameters, and regularization techniques. Each model iteration builds upon the previous one by addressing specific limitations and aiming for better performance metrics, as explained in sections IV-B and VI-B.

1) **Model 1** was designed in TensorFlow Keras. It leverages user and item embeddings, an attention mechanism, and fully connected layers. We created this model in the following way:
   - **Input Layers:** The model was initialized with two input layers, one for user IDs and another for item IDs, each represented as a single integer.
   - **Embedding Layers:** User and item IDs were mapped to dense vectors using embedding layers with an embedding dimension specified by **embedding_dim**. Note, in section IV-B, Experiment 1 and Experiment 2 were performed on **Model 1** with

embedding dimension of 50 and 20, respectively. L2 regularization was applied to these embeddings to prevent overfitting.

- **Flattening:** The output embeddings from both the user and item embedding layers were flattened to convert them into single-dimensional vectors.
- **Element-wise Multiplication:** The newly flattened user and item vectors were multiplied element-wise to capture the interaction between users and items.
- **Reshape for Attention:** The interaction vector was reshaped to a 3-dimensional tensor to make it compatible with the attention layer.
- **Attention Mechanism:** An attention layer was applied to the reshaped interaction tensor, allowing the model to focus on the most relevant parts of the interaction data. The attention output was then flattened back into a single-dimensional vector.
- **Dense Layers:** The attention-enhanced interaction vector was passed through two dense, fully connected layers with 128 and 64 neurons, respectively, each followed by a *ReLU* activation function and L2 regularization.
- **Output Layer:** The final layer was a single neuron without an activation function, which predicted the rating as a continuous value.

The model was compiled with the Adam optimizer and Mean Squared Error (MSE) loss function, optimizing it for the task of rating prediction.

Layer (type)	**Output Shape**	**Param #**	**Connected to**
input_layer (InputLayer)	(None, 1)	0	-
input_layer_1 (InputLayer)	(None, 1)	0	-
embedding (Embedding)	(None, 1, 50)	2,694,400	input_layer[0][0]
embedding_1 (Embedding)	(None, 1, 50)	11,450,650	input_layer_1[0][0]
flatten (Flatten)	(None, 50)	0	embedding[0][0]
flatten_1 (Flatten)	(None, 50)	0	embedding_1[0][0]
multiply (Multiply)	(None, 50)	0	flatten[0][0], flatten_1[0][0]
reshape (Reshape)	(None, 1, 50)	0	multiply[0][0]
attention (Attention)	(None, 1, 50)	0	reshape[0][0], reshape[0][0]
flatten_2 (Flatten)	(None, 50)	0	attention[0][0]
dense (Dense)	(None, 128)	6,528	flatten_2[0][0]
dense_1 (Dense)	(None, 64)	8,256	dense[0][0]
dense_2 (Dense)	(None, 1)	65	dense_1[0][0]

TABLE III: A detailed summary of the NCF-Attention model's architecture for Model 1 with 50 embedding layers copy

Layer (type)	**Output Shape**	**Param #**	**Connected to**
input_layer (InputLayer)	(None, 1)	0	-
input_layer_1 (InputLayer)	(None, 1)	0	-
embedding (Embedding)	(None, 1, 20)	1,077,760	input_layer[0][0]
embedding_1 (Embedding)	(None, 1, 20)	4,580,240	input_layer_1[0][0]
flatten (Flatten)	(None, 20)	0	embedding[0][0]
flatten_1 (Flatten)	(None, 20)	0	embedding_1[0][0]
multiply (Multiply)	(None, 20)	0	flatten[0][0], flatten_1[0][0]
reshape (Reshape)	(None, 1, 20)	0	multiply[0][0]
attention (Attention)	(None, 1, 20)	0	reshape[0][0], reshape[0][0]
flatten_2 (Flatten)	(None, 20)	0	attention[0][0]
dense (Dense)	(None, 128)	2,688	flatten_2[0][0]
dense_1 (Dense)	(None, 64)	8,256	dense[0][0]
dense_2 (Dense)	(None, 1)	65	dense_1[0][0]

TABLE IV: A detailed summary of the NCF-Attention model's architecture for Model 1 with 20 embedding layers

2) **Model 2** enhances **Model 1** with several improvements in order to potentially enhance both predictive performance and model robustness. Firstly, this architecture now incorporates deeper layers with increased neuron counts in the dense layers. Specifically, the dense layers were expanded to include 512, 256, 128, and 64 neurons sequentially, each utilizing *ReLU* activation and L2 regularization in order to promote deeper feature extraction and model complexity management. Additionally, dropout layers with a rate of 0.3 were introduced after each dense layer. This aimed to mitigate overfitting by randomly dropping 30% of the neurons during training, thus, encouraging the network to learn more robust and generalizable representations. Furthermore, the learning rate of the Adam optimizer was adjusted to 0.001 to potentially facilitate smoother convergence during training.

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 1)	0	-
input_layer_1 (InputLayer)	(None, 1)	0	-
embedding (Embedding)	(None, 1, 50)	2,694,400	input_layer[0][0]
embedding_1 (Embedding)	(None, 1, 50)	11,450,600	input_layer_1[0][0]
flatten (Flatten)	(None, 50)	0	embedding[0][0]
flatten_1 (Flatten)	(None, 50)	0	embedding_1[0][0]
multiply (Multiply)	(None, 50)	0	flatten[0][0], flatten_1[0][0]
reshape (Reshape)	(None, 1, 50)	0	multiply[0][0]
attention (Attention)	(None, 1, 50)	0	reshape[0][0], reshape[0][0]
flatten_2 (Flatten)	(None, 50)	0	attention[0][0]
dense (Dense)	(None, 512)	26,112	flatten_2[0][0]
dropout_1 (Dropout)	(None, 512)	0	dense[0][0]
dense_1 (Dense)	(None, 256)	131,328	dropout_1[0][0]
dropout_2 (Dropout)	(None, 256)	0	dense_1 [0][0]
dense_2 (Dense)	(None, 128)	32,896	dropout_2 [0][0]
dropout_3 (Dropout)	(None, 128)	0	dense_2 [0][0]
dense_3 (Dense)	(None, 64)	8,256	dropout_3[0][0]
dropout_4 (Dropout)	(None, 64)	0	dense_3[0][0]
dense_4 (Dense)	(None, 1)	65	dropout_4 [0][0]

TABLE V: A detailed summary of the NCF-Attention model's architecture for Model 2

3) **Model 3** builds upon and enhances **Model 1** and **Model 2** in the following ways. Firstly, the dense layers were augmented with increased depth and regularization. Specifically, this model had dense layers with 512, 256, 128, and 64 neurons respectively, each utilizing a combination of L1 and L2 regularization with coefficients set to 0.01. This regularization strategy aims to constrain the model's complexity and prevent overfitting by penalizing large weights, thus, encouraging sparsity in the learned parameters. Moreover, *LeakyReLU* activation functions were employed after each dense layer to introduce non-linearity and allow a small gradient when the unit is not active. This could potentially further help the model to capture complex relationships within the data. Additionally, dropout layers with a rate of 0.4 were implemented after each dense layer to combat overfitting by randomly dropping out 40% of the neurons during training. Overall, the goal of **Model 3** was to improve upon the generalization to unseen data, as well as the overall predictive accuracy in rating prediction tasks of **Model 1** and **Model 2**.

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 1)	0	-
input_layer_1 (InputLayer)	(None, 1)	0	-
embedding (Embedding)	(None, 1, 100)	5,388,800	input_layer[0][0]
embedding_1 (Embedding)	(None, 1, 100)	22,901,300	input_layer_1[0][0]
flatten (Flatten)	(None, 100)	0	embedding[0][0]
flatten_1 (Flatten)	(None, 100)	0	embedding_1[0][0]
multiply (Multiply)	(None, 100)	0	flatten[0][0], flatten_1[0][0]
reshape (Reshape)	(None, 1, 100)	0	multiply[0][0]
attention (Attention)	(None, 1, 100)	0	reshape[0][0], reshape[0][0]
flatten_2 (Flatten)	(None, 100)	0	attention[0][0]
dense (Dense)	(None, 512)	51,712	flatten_2[0][0]
leaky_re_lu_1 (LeakyReLU)	(None, 512)	0	dense[0][0]
dropout_1 (Dropout)	(None, 512)	0	leaky_re_lu_1[0][0]
dense_1 (Dense)	(None, 256)	131,328	dropout_1[0][0]
leaky_re_lu_2 (LeakyReLU)	(None, 256)	0	dense_1[0][0]
dropout_2 (Dropout)	(None, 256)	0	leaky_re_lu_2 [0][0]
dense_2 (Dense)	(None, 128)	32,896	dropout_2 [0][0]
leaky_re_lu_3 (LeakyReLU)	(None, 128)	0	dense_2[0][0]
dropout_3 (Dropout)	(None, 128)	0	leaky_re_lu_3 [0][0]
dense_3 (Dense)	(None, 64)	8,256	dropout_3[0][0]
leaky_re_lu_4 (LeakyReLU)	(None, 64)	0	dense_3[0][0]
dropout_4 (Dropout)	(None, 64)	0	leaky_re_lu_4[0][0]
dense_4 (Dense)	(None, 1)	65	dropout_4 [0][0]

TABLE VI: A detailed summary of the NCF-Attention model's architecture for Model 3

D. Book Recommendation Process

In order to recommend books to a user, we first have to select a specific user. Note, in this paper, we used user ID 3885, as seen in section V, due to this user's substantial rating history of 189 books. We anticipated that having a bigger dataset would give a bigger variety of book recommendations, as well as more interesting results for our research. We retrieved user ID 3885's interactions from the validation dataset in order to identify which books they have already interacted with. Next, we determined the books with which the user hasn't interacted (i.e., rated), by finding the difference between all items in the validation set and the items the user has already interacted with. For these non-interacted books, we created prediction inputs consisting of the user ID repeated for each item and the respective item IDs. Using one of our trained models, we predicted ratings for these items. After obtaining the predicted ratings, we combined the item IDs with their predicted ratings into a DataFrame. Then, we sorted it in descending order of predicted ratings in order to find the top-rated books. Finally, we extracted the titles and authors of the top 10 recommended books from the items DataFrame, and displayed them as the top recommendations for the user. This method ensures that the recommendations were personalized based on the user's past interactions and our model's predictions.

IV. Experiments

For all experiments, training progress and validation were assessed using training data comprising user IDs (**user_ids_train**), item IDs (**item_ids_train**), and their corresponding ratings (**ratings_train**). While validation data (**user_ids_val**, **item_ids_val**, **ratings_val**) was used to evaluate model performance during training.

A. Matrix Factorization with Neural Networks (MF-NN)

During our very first experiment, we observed significant overfitting in the model. To address this, we introduced L2 regularization across the embedding and dense layers. This approach aimed to enhance generalization performance by reducing the complexity of learned parameters, thus, improving the model's ability to generalize to unseen data.

1) ***Experiment 1****:* In this experiment, an embedding dimension of 50 was selected to define the size of the vectors representing user and item IDs in the model. The training process utilized early stopping with a patience of 3 epochs to monitor validation loss and restore the best weights when necessary in order to prevent overfitting. The model was trained over 10 epochs using a batch size of 32.

2) ***Experiment 2****:* In this experiment, an embedding dimension of 20 was chosen to define the vector size for representing user and item IDs in the model. To manage training progress and prevent overfitting, early stopping with a patience of 3 epochs was implemented. This approach continuously monitored validation loss and restored the model to its best weights observed during training. The model was trained over 10 epochs with a batch size of 32.

3) ***Experiment 3****:* In this experiment, an embedding dimension of 50 was employed to define the vector size for representing user and item IDs in the model. To optimize training dynamics and enhance model performance, two callbacks were utilized. The first callback, *ReduceLROnPlateau*, adjusted the learning rate dynamically by reducing it by a factor of 0.2 if no improvement in validation loss was observed for 2 consecutive epochs. This ensured smoother convergence towards the optimal solution with a minimum learning rate constraint of 0.001. The second callback, *EarlyStopping*, monitored validation loss and halted training if no improvement was detected for 3 consecutive epochs. This restored the model to its best weights. The model was trained over 50 epochs with a batch size of 32.

4) ***Experiment 4****:* In this experiment, an embedding dimension of 20 was chosen to define the vector size for representing user and item IDs in the model. To enhance training efficiency and performance, two callbacks were implemented. The first callback, *ReduceLROnPlateau*, dynamically adjusted the learning rate by reducing it by a factor of 0.2 if no improvement in validation loss was observed for 2 consecutive epochs. This ensured smoother convergence towards the optimal solution with a minimum learning rate constraint of 0.001. The second callback, *EarlyStopping*, monitored validation loss and halted training if no improvement was detected for 3 consecutive epochs. This restored the model to its best weights. The model was trained over 50 epochs with a batch size of 32.

B. Neural Collaborative Filtering with Attention (NCF-Attention)

1) ***Experiment 1****:* In this experiment, we tested **Model 1** with an embedding dimension of 50, which defined the vector size for representing user and item IDs in the model. Early stopping with a patience of 3 was employed to monitor validation loss and halt training if no improvement was observed over 3 consecutive epochs. This aimed to prevent overfitting. The model was trained for 10 epochs with a batch size of 32.

2) ***Experiment 2****:* In this experiment, we tested **Model 1** with an embedding dimension of 20, which defined the vector size for representing user and item IDs in the model. Early stopping with a patience of 3 was employed to monitor validation loss and halt training if no improvement was observed over 3 consecutive epochs. This aimed to prevent overfitting. The model was trained for 10 epochs with a batch size of 32.

3) ***Experiment 3****:* In this experiment, we tested **Model 2** with an embedding dimension of 50, which defined the vector size for representing user and item IDs in the model. Early stopping with a patience of 5 was employed to monitor validation loss and halt training if no improvement was observed over 5 consecutive epochs. Additionally, learning rate reduction was implemented using the *ReduceLROnPlateau* callback, which dynamically adjusted the learning rate by a factor of 0.5 if validation loss plateaued for 2 consecutive epochs. This ensured smoother convergence towards the optimal solution with a minimum learning rate constraint of 0.00001. The model underwent training for 30 epochs using a batch size of 512.

4) ***Experiment 4****:* In this experiment, we tested **Model 3** with an embedding dimension of 100, which defined the vector size for representing user and item IDs in the model. Early stopping was implemented with a patience of 10 epochs, enabling the model to halt training when validation loss ceased to improve over 10 consecutive epochs. This aimed to prevent overfitting and restore weights corresponding to the best validation performance. Additionally, learning rate reduction using *ReduceLROnPlateau* dynamically adjusted the learning rate by a factor of 0.5 if validation loss plateaued for 3 consecutive epochs. This ensured smoother convergence towards the optimal solution with a minimum learning rate constraint of 0.00001. The model was trained over 50 epochs using a batch size of 256.

C. Book Recommendation for a Specific User

We tested our book recommendation process, as explained in section III-D, on several randomly chosen users from the dataset. In this paper, we report results in section V for User ID 3885 because this user had rated a substantial number of books, making the book recommendations from our models more interesting and insightful to discuss in section VI.

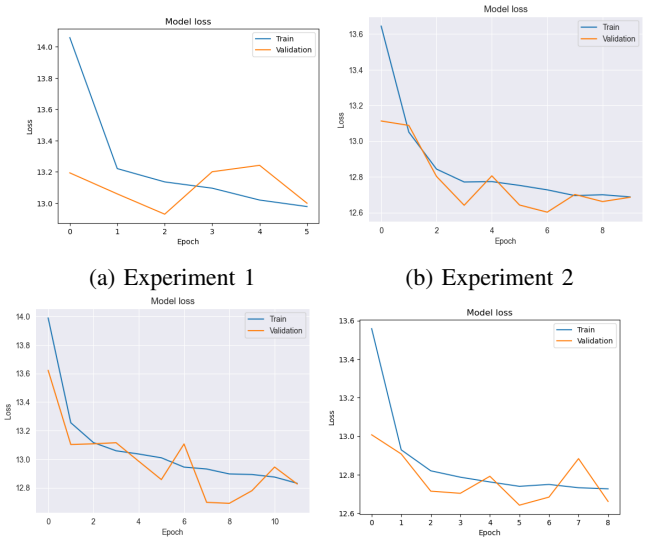
V. Results

A. Specific Example: Book Recommendations for User 3885

Book Title	Author
A Hiker's Companion: 12,000 Miles of Trail-Tested Wisdom	Cindy Ross
Pet Sematary	Stephen King
The Murders of Richard III	Elizabeth Peters
Night Prey	Carol Davis Luce
Black Livingstone: A True Tale of Adventure in the Nineteenth-Century Congo	Pagan Kennedy
Tales from the Dark Tower	Joseph Vargo
Els intel·lectuals, avui (Treballs de la Secció de Filosofia i Ciències Socials)	Jordi Berrio
French Lessons: A Memoir	Alice Kaplan
The Cat Who Tailed a Thief	Lilian Jackson Braun
She'S Having His Baby (Accidental Dads) (Harlequin American Romance, 751)	Linda Randall Wisdom

TABLE VII: Top 10 highest-rated books by user 3885

B. Matrix Factorization with Neural Networks (MF-NN)



(a) Experiment 1 (b) Experiment 2

(c) Experiment 3 (d) Experiment 4

Fig. 1: Training and Validation loss values of MF-NN from each experiment

	Test loss	Validation loss	Validation RMSE	Test RMSE
Experiment 1	12.970	12.929	3.407	3.413
Experiment 2	12.608	12.602	3.412	3.413
Experiment 3	12.715	12.691	3.409	3.412
Experiment 4	12.672	12.642	3.412	3.416

TABLE VIII: Comparison of Test and Validation metrics of MF-NN across experiments

Book Title	Author
Garzanti - Gli Elefanti: Creatura Di Sabbia	T Ben Jelloun
Afrikanisches Fieber: Erfahrungen aus vierzig Jahren	Ryszard Kapuscinski
I And Thou	Martin Buber
A Grave Denied: A Kate Shugak Novel	Dana Stabenow
Buddha of Suburbia	Hanif Kureishi
Dana and the Calendar Man (Harlequin American)	Hiram Bingham
Three Junes	Julia Glass
Flood: Mississippi 1927	Kathleen Duey
How to Talk So Kids Will Listen & Listen So Kids Will Talk (How To Talk Series)	Adele Faber
Life Penalty	Joy Fielding

TABLE IX: MF-NN Model 1's Top 10 recommended books for user 3885

Book Title	Author
How to Talk So Kids Will Listen & Listen So Kids Will Talk (How To Talk Series)	Adele Faber
Afrikanisches Fieber: Erfahrungen aus vierzig Jahren	Ryszard Kapuscinski
The Luberon Garden	Alex Dingwall-Main
Garzanti - Gli Elefanti: Creatura Di Sabbia	T Ben Jelloun
Life Penalty	Joy Fielding
Dana and the Calendar Man (Harlequin American)	Hiram Bingham
Timeless/Four Breathtaking Tales of Hearts	Linda Lael Miller
Tapestry Of Fate (Harlequin Historical, No 246)	Nina Beaumont
Gdje pijevac ne pjeva (Biblioteka Zlatni paun)	Ivan Aralica
Vieja Sirena, La	Jose Luis Sampedro

TABLE X: MF-NN Model 2's Top 10 recommended books for user 3885

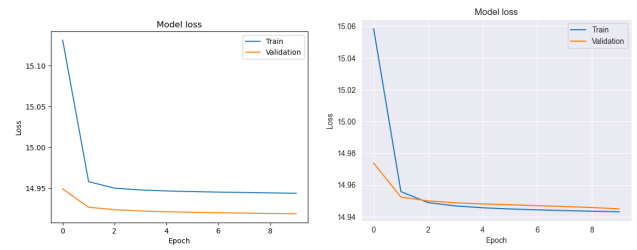
Book Title	Author
Garzanti - Gli Elefanti: Creatura Di Sabbia	T Ben Jelloun
Afrikanisches Fieber: Erfahrungen aus vierzig Jahren	Ryszard Kapuscinski
Dana and the Calendar Man (Harlequin American)	Hiram Bingham
I Capture the Castle	Dodie Smith
Flood: Mississippi 1927	Kathleen Duey
Life Penalty	Joy Fielding
Tapestry Of Fate (Harlequin Historical, No 246)	Nina Beaumont
A Grave Denied: A Kate Shugak Novel	Dana Stabenow
I And Thou	Martin Buber
The Diary of Jack the Ripper: The Discovery, the Investigation, the Debate	Jack

TABLE XI: MF-NN Model 3's Top 10 recommended books for user 3885

Book Title	Author
Garzanti - Gli Elefanti: Creatura Di Sabbia	T Ben Jelloun
Tapestry Of Fate (Harlequin Historical, No 246)	Nina Beaumont
How to Talk So Kids Will Listen & Listen So Kids Will Talk (How To Talk Series)	Adele Faber
Afrikanisches Fieber: Erfahrungen aus vierzig Jahren	Ryszard Kapuscinski
A Grave Denied: A Kate Shugak Novel	Dana Stabenow
Hiding Place	Corrie Ten Boom
The Tao of Bada Bing: Words of Wisdom from the Sopranos	David Chase
Hot Off the Press	Nancy Warren
White Oleander: A Novel	Janet Fitch
Flood: Mississippi 1927	Kathleen Duey

TABLE XII: MF-NN Model 4's Top 10 recommended books for user 3885

C. Neural Collaborative Filtering with Attention (NCF-Attention)



(a) Experiment 1 (b) Experiment 2

(c) Experiment 3 (d) Experiment 4

Fig. 2: Training and Validation loss values of NCF-Attention from each experiment

	Test loss	Validation loss	Validation RMSE	Test RMSE
Experiment 1	14.948	14.918	3.862	3.866
Experiment 2	14.957	14.945	3.865	3.867
Experiment 3	14.940	14.940	3.865	3.865
Experiment 4	14.958	14.958	3.866	3.867

TABLE XIII: Comparison of Test and Validation metrics of NCF-Attention across experiments

Book Title	Author
Decision in Normandy	Carlo D'Este
Lady and the Outlaw	Joyce Brandon
Girls Out Late	Jacqueline Wilson
Of Beetles and Angels: A Boy's Remarkable Journey from a Refugee Camp to Harvard	Mawi Asgedom
O Is for Outlaw (Kinsey Millhone Mysteries)	Sue Grafton
Hug a Teddy	Jim. Erskine
The Asimov Chronicles: Fifty Years of Isaac Asimov	Isaac Asimov
An Unsuitable Job for a Woman (BBC Radio)	P. D. James
Bountiful Women: Large Women's Secrets for Living the Life They Desire	Bonnie Bernell
Evolution's End: Claiming the Potential of Our Intelligence	Joseph C. Pearce

TABLE XIV: NCF-Attention Model 1's Top 10 recommended books for user 3885

Book Title	Author
Classical Mythology	Mark P. O. Morford
Bountiful Women: Large Women's Secrets for Living the Life They Desire	Bonnie Bernell
Kurt Cobain	Kurt Cobain
La méthode simple pour en finir avec la cigarette	Allen Carr
Easy Cooking for One or Two	Louise Davies
Nightfire (Heartfire)	Leona Karr
Starlight (Heartfire Romance)	Gloria Dale Skinner
The Old Contemptibles-O.M.	Martha Grimes
The Bargain	Jane Ashford
The World of Jeeves	P.G. Wodehouse

TABLE XV: NCF-Attention Model 2 and Model 3 recommend the same Top 10 recommended books for user 3885

Book Title	Author
Decision in Normandy	Carlo D'Este
Lady and the Outlaw	Joyce Brandon
Girls Out Late	Jacqueline Wilson
Of Beetles and Angels: A Boy's Remarkable Journey from a Refugee Camp to Harvard	Mawi Asgedom
O Is for Outlaw (Kinsey Millhone Mysteries)	Sue Grafton
Hug a Teddy	Jim. Erskine
The Asimov Chronicles: Fifty Years of Isaac Asimov	Isaac Asimov
An Unsuitable Job for a Woman (BBC Radio)	P. D. James
Bountiful Women: Large Women's Secrets for Living the Life They Desire	Bonnie Bernell
Evolution's End: Claiming the Potential of Our Intelligence	Joseph C. Pearce

TABLE XVI: NCF-Attention Model 4's Top 10 recommended books for user 3885

VI. Discussion

A. Matrix Factorization with Neural Networks (MF-NN)

1) ***Model 1****:* The training process showed an initial decline in both training and validation loss, suggesting that the model was learning from the data. However, it displayed overfitting as the validation loss decreased up to the second epoch and then started increasing in the next epochs. This suggests that while the model started to capture the patterns in the data, it failed to generalize well, and instead ended up memorizing the training data. The final test and validation RMSE values indicate that the model's predictive accuracy is moderate but suffers from overfitting issues, limiting its performance on unseen data.

2) ***Model 2****:* Model 2 showed a more stable and consistent decrease in both training and validation loss throughout most of the training process. It generalized better than Model 1, as evidenced by the gradual decrease in validation loss. Even though there were slight fluctuations in the later epochs, it generally decreased alongside the training loss. This trend suggests that Model 2 managed to capture the underlying data patterns without overfitting significantly. The final, almost identical test and validation RMSE values reflect stable and reliable predictive performance, indicating good generalization capabilities and making Model 2 a potentially robust model for recommendations.

3) ***Model 3****:* Initially, Model 3 showed a steady decrease in both training and validation loss, similar to Model 1, but with a more extended training period. While, at first, the validation loss showed improvement and stability, there were some fluctuations, which were more pronounced compared to Model 2. This suggests overfitting tendencies. Even though, the test loss and validation loss are slightly better than Model 1, they are slightly higher than those in Model 2. This indicates that while the model performs well, it may not be the best among the four experiments.

4) ***Model 4****:* Model 4 showed a stable decrease in both training and validation loss, similar to Model 2. The validation loss closely followed the training loss, indicating good generalization. There were some fluctuations, but they were minor and did not indicate significant overfitting. This model seems to generalize well, but the slightly higher test loss, validation loss, and test RMSE values suggest that it might not be as accurate as Model 2.

5) ***The Best MF-NN model****:* Comparing these results, Model 2 would be the best performing MF-NN model out of the four proposed models. It showed consistent improvement in validation loss with minimal fluctuations, indicating good generalization without significant overfitting nor underfitting. While Model 4 also performed well, its slightly higher test loss, validation loss, and test RMSE values suggest it is marginally less accurate than Model 2.

B. Neural Collaborative Filtering with Attention (NCF-Attention)

1) ***Model 1****:* Model 1 showed a rapid initial decrease of the training and validation losses, followed by their stabilization. Both losses were very close to each other throughout the training process. This indicates that the model generalizes well without significant overfitting nor underfitting. The test and validation loss values, as well as the RMSE values are relatively low and quite similar, which suggests that the model is making accurate predictions. This consistency indicates that the model is learning effectively from the data and generalizes well to unseen data.

2) ***Model 2****:* In Model 2, both training and validation losses started from higher values compared to Model 1, yet had a similar decreasing trend, followed by a stabilization. Both losses were almost identical throughout the epochs, indicating good generalization without overfitting. The test and validation loss values, along with the RMSE values, are slightly higher than in Model 1, but still show strong performance. Additionally, in Model 2, the validation set performs better than the training set over the epochs, compared to Model 1.

3) ***Model 3****:* In Model 3, the training loss started significantly higher than the other models, which might suggest that initially, it had more difficulty fitting the data. However, the rapid decrease in loss over epochs indicates that the model began to learn effectively. Towards the later epochs, the rate of improvement in validation loss slowed down, suggesting that further training may not significantly improve performance without adjustments. Additionally, the validation set seems to perform worse than the training set over the epochs. The values of the test loss and validation loss are identical, as well as the lowest among all four experiments. Both RMSE values are identical as well. This indicates a well-performing and generalizing model.

4) ***Model 4****:* Model 4 showed signs of overfitting as the training loss initially decreased much faster than the validation loss. The initial large difference between training and validation losses indicates that the model may have learned specific features of the training data that do not generalize well to new data. After several epochs, the losses stabilized and became very close to each other throughout the epochs. This indicated good generalization. However, the final test and validation loss values, as well as the RMSE values, are the highest among the four experiments. This suggests that while the model generalizes well, it is the least accurate compared to the other experiments.

5) ***The best NCF-Attention model****:* Comparing these results, Model 1 and Model 2 show the most consistent improvement in both training and validation losses without significant signs of overfitting nor underfitting. However, we deem Model 2 as the best performing NCF-Attention model out of the four proposed models because the validation set performs better than the training set over the epochs.

C. *Comparison between the best-performing MF-NN model and the best-performing NCF-Attention*

Model 2 of MF-NN demonstrates better accuracy with lower test and validation losses compared to Model 2 of NCF-Attention. The RMSE values are also lower for Model 2 of MF-NN, indicating more accurate predictions. While both

models are capable of generalizing well, Model 2 of MF-NN stands out as the better model overall due to its higher accuracy and lower error rates. This model would be more likely to provide more accurate and reliable book recommendations to users based on their reading history. Therefore, leveraging traditional collaborative filtering approaches with the added flexibility of neural networks, as we did with the MF-NN model, proves to be a more suitable method for our dataset.

D. Comparison of the book recommendations of each best-performing model for User 3885

Model 2 of MF-NN Model recommends a mix of educational, non-fiction, and fiction titles, aligning closely with user 3885's preferences and reading history. In contrast, Model 2 of NCF-Attention recommends more classical literature and personal development books, which may not align as closely with the user's previous interests. This aligns with the quantitative data, indicating that Model 2 of MF-NN has superior personalization capabilities compared to the best-performing NCF-Attention model. Therefore, Model 2 of MF-NN is once again the preferred choice for book recommendations for this dataset.

VII. Limitations and Future Work

We acknowledge some potential limitations in our work. Even though the dataset we used is extensive, it might not fully represent the diversity of user preferences and interactions in a real-world scenario. This might lead to inaccuracies in our findings. Furthermore, we did not conduct any experiments involving the "cold start" problem as we determined that new users and books without sufficient number of ratings did not align with our research agenda. Therefore, we cannot predict how our models would perform in such scenarios.

We anticipate future work to entail handling of "cold start" scenarios. This would involve developing strategies to make accurate predictions for new users and items with sparse rating histories. For further investigation, we also propose comparison between our models and state-of-the-art models. Such comparisons would provide deeper insights into the strengths and weaknesses of our proposed models.

VIII. Conclusion

In this paper, we provided a comprehensive comparison of two hybrid book recommender models: Matrix Factorization with Neural Networks (MF-NN) and Neural Collaborative Filtering with Attention (NCF-Attention). Our extensive experiments on the "Book-Crossing: User review ratings" dataset revealed that MF-NN offers better accuracy and generalization compared to NCF-Attention. The main takeaway is that MF-NN provides a more balanced approach between accuracy and computational efficiency, making it a more suitable choice for our dataset.

References

[1] Mohammed Fadhel Aljunid and Manjaiah Dh. An efficient deep learning approach for collaborative filtering recommender system. *Procedia Computer Science*, 171:829–836, 2020. Third International Conference on Computing and Network Communications (CoCoNet'19).

[2] Ruchi Bhatia. Book-crossing: User review ratings. https://www.kaggle.com/datasets/ruchi798/bookcrossing-dataset/data, 2021.

[3] Hai Chen, Fulan Qian, Jie Chen, Shu Zhao, and Yanping Zhang. Attribute-based neural collaborative filtering. *Expert Systems with Applications*, 185:115539, 2021.

[4] Xin Dong, Lei Yu, Zhonghuo Wu, Yuxia Sun, Lingfeng Yuan, and Fangxi Zhang. A hybrid collaborative filtering model with deep structure for recommender systems. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 1309–1315. AAAI Press, 2017.

[5] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[6] Yanxia Lv, Ying Zheng, Fangna Wei, Cong Wang, and Cuirong Wang. Aicf: Attention-based item collaborative filtering. *Advanced Engineering Informatics*, 44:101090, 2020.

[7] Conor Murray. Tiktok is taking the book industry by storm, and retailers are taking notice.

[8] Nour Nassar, Assef Jafar, and Yasser Rahhal. Multi-criteria collaborative filtering recommender by fusing deep neural network and matrix factorization. *Journal of Big Data*, 7(1):34, May 2020.

[9] Ruiqin Wang, Zongda Wu, Jungang Lou, and Yunliang Jiang. Attention-based dynamic user modeling and deep collaborative filtering recommendation. *Expert Systems with Applications*, 188:116036, 2022.

[10] Hongbin Xia, Yang Luo, and Yuan Liu. Attention neural collaboration filtering based on gru for recommender systems. *Complex & Intelligent Systems*, 7:1367 – 1379, 2021.

[11] Hong-Jian Xue, Xin-Yu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. Deep matrix factorization models for recommender systems. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 3203–3209. AAAI Press, 2017.

[12] Krishan Kant Yadav, Hemant Kumar Soni, Ghanshyam Yadav, and Mamta Sharma. Collaborative filtering based hybrid recommendation system using neural network and matrix factorization techniques. *International Journal of Intelligent Systems and Applications in Engineering*, 12(8s):695–701, Dec. 2023.

[13] Chao Yang, Lianhai Miao, Bin Jiang, Dongsheng Li, and Da Cao. Gated and attentive neural collaborative filtering for user generated list recommendation. *Knowledge-Based Systems*, 187:104839, 2020.

[14] Jim Zarroli. Tiktok is driving book sales. here are some titles booktok recommends.