# CHAPTER TWO

# LITERATURE REVIEW

## 2. 0 Introduction

The purpose of this literature review is to critically evaluate the existing research on deep learning models for personalized book recommendations. The aim is to identify the strengths and limitations of current approaches, highlight the research gaps and challenges and provide a basis for the development of a novel deep learning model for personalized book recommendation.

## 2.1 Book Recommendation Systems

Book recommendation systems are software applications that suggest books to users based on their interests and preferences. The goal of these systems is to help users discover books that they might be interested in reading but might not otherwise have discovered on their own.

There are two types of recommendation systems i.e., personalized and non-personalized recommendation system. Personalized recommendation systems recommend product or items to users based on preference criteria of individual (Sana, James & Nasseh, 2019). This system makes use of the individual past records of user’s ratings in order to find more suitable items that are yet to be rated by the user as recommendations (Khalid & Jamshed, 2020). On the other hand, non-personalized recommendation systems recommend base on the current popular trends that are relevant to all users. These trends can be the top 10 most popular items among users, information on a certain update or the current updates of a particular environment which are suitable for every new user (Khatwani & Chandak, 2016).

In recent years, deep learning models have become increasingly popular for book recommendations (Dhanashri et al., 2020). These models use artificial neural networks and other machine learning techniques to process large amounts of data and make recommendations based on patterns and trends in the data (Balaji, Pranshu & Deepali, 2020). Deep learning models can incorporate a wide range of data, including user behavior, book metadata, and social network information, to make highly personalized recommendations (Alexandros & Balázs, 2017).

However, there are several challenges associated with developing effective book recommendation systems. These include the cold start problem, which occurs when there is not enough user data to make accurate recommendations for new users, and the problem of data sparsity, which occurs when there is not enough data on certain books or users to make accurate recommendations. Additionally, privacy concerns related to user data collection and usage are an important issue that must be addressed when developing book recommendation systems.

This brief overview of book recommendation systems sets the stage for the subsequent sections of the literature review, which will dive deeper into the different types of recommendation systems and the challenges associated with developing effective deep learning models for personalized book recommendations.

## 2.2 Personalized Recommendation system

The emergence of personalized recommendation system can be traced back to the extensive research in cognitive science, approximation theory, information retrieval, forecasting theories, and also to management science related and to consumer choice modeling in marketing i.e., E-commerce. The recommender systems emerged as an autonomous research domain in the mid-1990s. This is the period where researchers started to focus on the recommendation problems that rely solely on ratings structure. The recommendation problem is reduced to the problem of recommender system to be estimating ratings for the items that have not been seen or rated by the user. Intuitively, this prediction or estimation is usually based on the user’s previous rated items. In this sense, if the estimate ratings for the yet unrated items can be obtain, then the item(s) with the highest estimated rating(s) can be predicted to the user (Gediminas, Alexander et al., 2005).

Personalized recommendation system is a computer-based system that offers individualized recommendations to users based on their previous behavior or history (Khatwani & Chandak, 2016). According to Dina F. et al. (2018), a recommender system in the context of education provides intelligent recommendations to students about study materials, courses, and related information based on their prior decisions. To provide such recommendations, the system employs a statistical model that is trained using a learning algorithm based on input from previous learning participants.

In a personalized recommendation system, users and items are the two main objects that play a crucial role. Users provide their interests about items, and this information is accumulated as input data in a utility matrix. The utility matrix represents the order of preference for particular items by the user in the combination of customer-item value. There are two classifications of recommender systems: user-based and item-based. In a user-based system, the user's interests, dislikes, and ratings are used to recommend unrated items. In contrast, an item-based system utilizes the relationship among items to generate recommendations for users (Geetha & Karthika, 2019).

Personalized recommendation systems play a vital role in both academia and industries. Many companies use personalized RS in their sales promotion, utilizing various platforms. For instance, the majority of the most-watched movies on YouTube and other online video databases come from the RS (Aminu & Naomie, 2019). The main purpose of building a recommender system is to provide maximum information required for the sole aim of personalizing learning and interests depending on the interactive patterns of users (Bouihi & Bahaj, 2019; Zhang & Yang, P, 2020).

The algorithm for estimating the not-yet-rated items in the personalized recommender system is classified into three, namely: collaborative filtering, content base and hybrid approach (Kunal, Akshaykumar et al., 2017).

All authors agree that a personalized recommendation system is a system created with the major aim of recommending useful individualized information or items to users based on past learned historical records relating to the individual’s tastes or preferences.

### 2.2.1 Type of recommendation systems

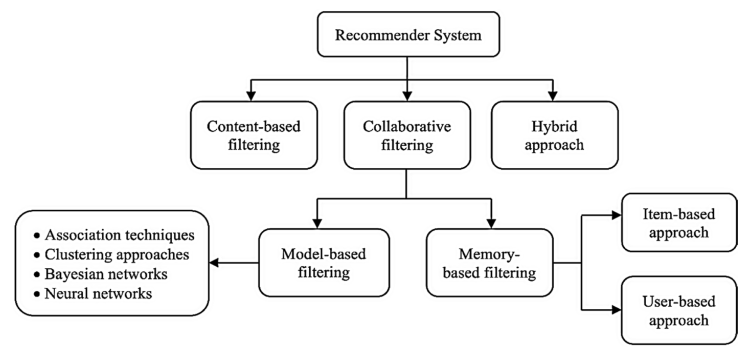
There are three main categories of recommendation systems: content-based recommender systems, collaborative recommender systems, and hybrid recommender systems (Roy & Dutta, 2022). Figure 1 provides a visual depiction of these different types of recommender systems.

### 2.2.2 Content-based filtering (CBF)

Content-based recommender systems are widely used in information or book retrieval. The process involves manually assigning terms and selecting appropriate techniques to compare these terms with the information in the user's profile. A learning algorithm is then applied to perform the chosen techniques and provide relevant recommendations to the user (Javed & Shaukat, 2021).

In content-based recommender systems, all the data items are collected into different item profiles based on their description or features. For example, in the case of a book, the features will be author, publisher, etc. When a user gives a positive rating to an item, then the other items present in that item profile are aggregated together to build a user profile. This user profile combines all the item profiles, whose items are rated positively by the user. Items present in this user profile are then recommended to the user (Roy & Dutta, 2022).

In their study, Balush et al. provide a comprehensive description of Content-Based Filtering (CBF) as an algorithmic approach for recommending items or books to users based on their past consumption. The CBF method analyzes the interests of new users by considering the features of the objects they have evaluated. This approach operates as a recommendation system tailored to specific keywords, where these keywords are used to describe the items. Consequently, the content-based recommendation system employs algorithms to suggest items that are similar to those enjoyed or currently being explored by the user (Balush, Vysotska & Albota, 2021).



*Fig. 1 Types of recommendation systems*

Term frequency (TF) and inverse document frequency (IDF) are important concepts used in information retrieval and content-based filtering systems, such as content-based recommenders. These concepts determine the relative importance of a document, article, news item, film, etc. Content-based filtering is a common approach in recommender system design, relying on item descriptions and user profiles. In a content-based recommender system, keywords are used to describe items, and a personal profile is constructed to suggest items that align with the user's preferences. In other words, these algorithms aim to recommend items similar to those that the user has liked or currently shows interest in. To achieve this, multiple candidate items are compared with items previously rated by the user, and the best matching items are recommended. Yannick et al. (2010) employed a content-based algorithm to filter and recommend the best results to users. The advantages of content-based recommender systems include transparency, independence, and recommendations for unclassified entities. However, they also have drawbacks such as limited serendipity, partial content analysis, and overspecialization (Adomavicius, Tuzhilin, et al., 2005).

By leveraging a content-based recommender system, recommendations can be generated based on the user's provided data or implicit interactions with the system. As the user provides more data or accepts suggested activities, the system becomes increasingly accurate in generating personalized recommendations. Compared to collaborative filtering systems, content-based approaches offer easier implementation and have been used in many projects due to their ability to generate highly relevant and transparent results. Content-based filtering strategies often employ techniques for content extraction, characterization, and building user and item profiles. However, these techniques have limitations, such as mismatches between user profile items and profile terms, leading to lower performance. Notable examples of content-based recommendations include Fab, which suggests web pages, and ELFI, which recommends financial information from a database (Lops, Gemmis & Semeraro, 2011).

Content-based filtering systems adapt their behavior to individual users by learning their preferences from previously relevant documents. The process involves comparing representations of item content with representations of user interests to find the most relevant items for each user. This task requires finding the best representation for both items (item profile) and users (user profile). A user profile represents a mapping of the user's real-world interests to a simplified domain model, while an item's profile describes its content using keywords or other means of representation. By enabling matching between user and item profiles, a common representation is established (Javed & Shaukat, 2021).

Content-based recommendation systems operate by analyzing a collection of documents and/or descriptions of items that have been previously rated by a user. These systems construct a model or profile of the user's interests based on the features of the rated objects (Lops, Gemmis & Semeraro, 2011). As described by Mladenic (2018), the profile serves as a structured representation of user interests and is utilized to recommend new and interesting items. The recommendation process primarily involves comparing the attributes of the user profile with the attributes of a content object (Khatwani & Chandak, 2016). This comparison results in a relevance judgment that indicates the user's level of interest in the object. A well-constructed profile that accurately reflects user preferences greatly enhances the effectiveness of an information retrieval process (Geetha & Renuka, 2019).

An example of the profile's utility is its application in filtering search results. By leveraging the user profile, it becomes possible to determine whether a user would be interested in a specific web page. If not, the system can prevent the display of that particular page (Lops, Gemmis & Semeraro, 2011).

#### 2.2.2.1 High-Level Architecture of Content-based Systems

Content-based Information Filtering (IF) systems require effective techniques for representing items and generating user profiles, as well as strategies for comparing the user profile with item representations (Adomavicius, Tuzhilin, et al., 2005). In their study, Lops et al. (2011) outlined a three-step recommendation process, with each step handled by a distinct component:

1. **Content Analyzer**: The Content Analyzer is responsible for processing information without a predefined structure, such as text, and extracting relevant structured information. Its main task is to represent the content of items (e.g., documents, web pages, news, product descriptions) from various sources in a format suitable for further processing. Through feature extraction techniques, the component analyzes data items and transforms their representation from the original information space to the target space, such as representing web pages as keyword vectors. The output of the Content Analyzer serves as input for the Profile Learner and Filtering Component.
2. **Profile Learner**: The Profile Learner module collects representative data on user preferences and generalizes this data to construct the user profile. Typically, machine learning techniques are employed to infer a model of user interests based on items previously liked or disliked. For example, in a web page recommender system, the Profile Learner can utilize a relevance feedback method (Rocchio, J., 1971), where vectors of positive and negative examples are combined to create a prototype vector representing the user profile. Training examples consist of web pages that received positive or negative feedback from the user.
3. **Filtering Component**: The Filtering Component leverages the user profile to suggest relevant items by comparing the profile representation with the representations of items to be recommended. This comparison results in a binary or continuous relevance judgment, often computed using similarity metrics. In the case of continuous relevance, a ranked list of potentially interesting items is generated. In the previous example, the matching process involves computing the cosine similarity between the prototype vector and the item vectors.

The recommendation process begins with the Content Analyzer, which applies techniques from Information Retrieval systems to process item descriptions from information sources. The Content Analyzer extracts features such as keywords, n-grams, and concepts from unstructured text to generate a structured representation of items stored in the Represented Items repository.

To construct and update the user profile for the active user (Ua) who requires recommendations, the system collects the user's reactions to items and records them in the Feedback repository. These reactions, known as annotations or feedback, along with the corresponding item descriptions, are used during the process of learning a model that predicts the relevance of newly presented items. Users can either provide explicit feedback, where they actively evaluate items, or implicit feedback, which is derived from monitoring and analyzing user activities (Lops, Gemmis & Semeraro, 2011).

Explicit feedback can be obtained through approaches such as like/dislike ratings, numerical ratings, or text comments. Implicit feedback methods assign a relevance score based on user actions, such as saving, discarding, printing, sounds, manipulation of physical objects, or bookmarking of items. While explicit feedback is straightforward, the adoption of numeric or symbolic scales can increase cognitive load. Implicit feedback methods do not require direct user involvement but may be subject to biases, such as interruptions during reading (Meddeb, Maraoui & Zrigui, 2021).

According to Nilashi et al. (2013), user feedback and ratings can take different forms, which include:

* **Numerical ratings**: These ratings are represented by numbers on either discrete or continuous scales. Discrete rating scales, such as the popular five-star system or Likert response scales used in questionnaires, provide a limited range for users to rate items. Continuous rating scales often involve sliders that users can adjust to indicate a specific value.
* **Binary rating scale**: This type of rating allows users to categorize items into two distinct classes, typically denoted as "like" or "dislike." For example, platforms like YouTube offer users the option to give movies a thumbs-up or thumbs-down rating.
* **Ordinal ratings**: Users are presented with a set of options such as "strongly agree," "neutral," "disagree," or "strongly disagree." They are then asked to choose the option that best represents their opinion about a particular item, typically through the use of questionnaires.
* **Unary rating**: Users can assign items to a single positive class using unary ratings. A prominent example is Facebook's "Like" button, which allows users to express their positive opinion about a post. Implicit unary ratings can also be inferred from user actions like purchasing products on a web shop or clicking links on a webpage. The absence of a rating indicates a lack of information connecting the user to the item, potentially because the purchase was made elsewhere.

These different types of ratings provide users with diverse ways to express their preferences and opinions about items.

The Profile Learner component utilizes a training set (TRa) specific to the active user (Ua), consisting of pairs ⟨Ik, Rk⟩, where Rk represents the rating provided by Ua for the item representation Ik. By employing supervised learning algorithms on the labeled item representations, the Profile Learner generates a predictive model known as the user profile. This model is stored in a profile repository and utilized by the Filtering Component for future recommendations. When presented with a new item representation, the Filtering Component compares its features to those in the user profile, predicting the user's interest. Typically, the Filtering Component incorporates strategies to rank potentially interesting items based on their relevance to the user profile. The top-ranked items form a recommendation list (La) that is presented to the user (Ua). As user preferences change over time, the user profile is continuously updated by incorporating up-to-date information. User feedback on the recommendations helps refine the learning process, creating a feedback-learning cycle that adapts to the dynamic nature of user preferences (Lops, Gemmis, & Semeraro, 2011).

In summary, Content-based Information Filtering systems involve the Content Analyzer for item representation, the Profile Learner for constructing user profiles, and the Filtering Component for suggesting relevant items based on user profiles. These components interact in a recommendation process that incorporates user feedback to adapt to evolving user preferences.

### 2.2.3 Collaborative Filtering (CF)

The term "collaborative filtering" was first introduced in 1992 by Goldberg et al., who proposed that considering human group relational preferences or involvements can enhance the effectiveness of information filtering when recommending items to users (Kunal, Akshaykumar, et al., 2017). This recommendation method aims to predict the utility of items for a specific user by leveraging the ratings given to those items by other similar users in the past (Gediminas, Alexanda, et al., 2005).

Collaborative filtering, as described by Kunal et al. (2017), is a recommendation system approach that generates recommendations for different users by actively comparing the preferences of an active user with those of other users who have rated similar products in the past. On the other hand, Sana et al. (2019) define collaborative filtering as a technique that predicts recommendations by identifying users with similar interests. Collaborative filtering is based on the assumption that individuals who have agreed on item evaluations in the past are likely to agree in the future as well.

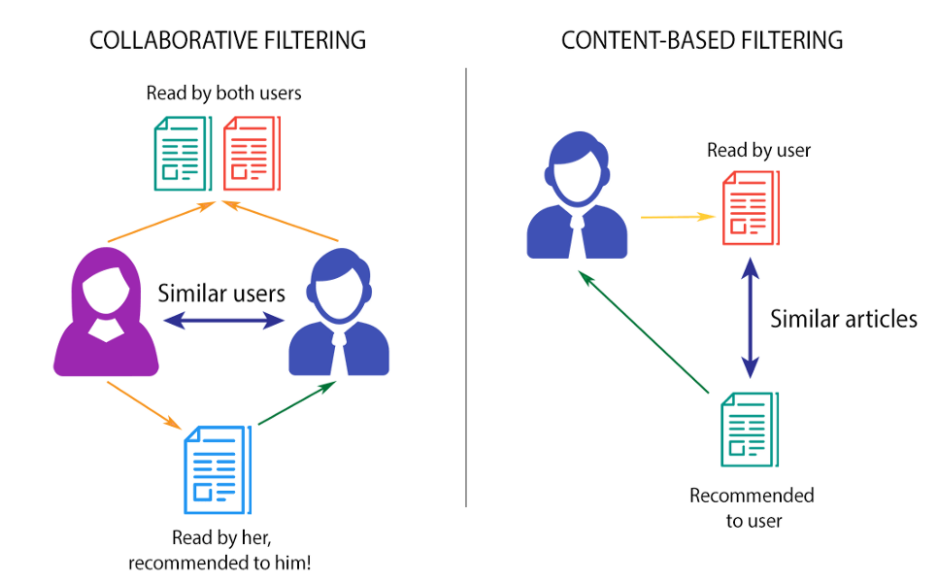
In accordance with Balush, Vysotska & Albota (2021), Collaborative Filtering (CF) is a type of recommendation system that relies on aggregating object ratings or recommendations, identifying similarities among users based on their ratings, and generating new recommendations through user comparisons. This approach proves effective in cases involving complex objects, where variations in taste account for the majority of preference changes. Collaborative filtering operates under the assumption that individuals who have agreed in the past will continue to have similar preferences in the future and are likely to enjoy similar objects as they have before.

Collaborative filtering method are grouped into two general classes: neighborhood based and model-based (Gediminas, Alexanda, et al., 2005; Kunal, Akshaykumar, et al., 2017).

#### 2.2.3.1 Neighborhood Based Method

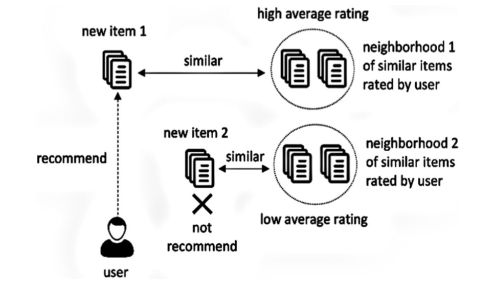
The neighborhood-based method of collaborative filtering, also known as memory-based or heuristic-based CF, leverages the user-item ratings stored in memory to make predictions about the preferences for new items. This can be accomplished through two approaches: user-based and item-based recommendation (Qian, Jie, et al., 2020).

In the user-based approach, the interest of a user (referred to as "u") in an item (referred to as "i") is determined by considering the ratings given to that item by other users who exhibit similar rating patterns. These users are known as "neighbors". By examining the ratings of these neighbors, the system can estimate the likelihood of user u being interested in item i (Dhanashri, Nandani, et al., 2020). This approach as explained by Roy & Dutta (2022), the user rating of a new item is calculated by finding other users from the user neighborhoods who has previously rated that same item. If a new item receives positive ratings from the user neighborhood, the new item is recommended to the user. Figure 2 depicts the user-based filtering approach.



*Fig. 2 User-based collaborative filtering*

On the other hand, the item-based approach involves predicting the rating that user (u) would assign to item (i) based on the ratings that user u has given to items similar to i. In this approach, the similarity between two items is calculated by examining the ratings provided by other users of the system. By analyzing these ratings, the system can identify items that are similar in terms of user preference and use this information to make predictions about user u's rating for item i (Bushra, Imran, et al., 2019). Also, Roy & Dutta (2022) addresses the fact that, in the item-based approach, an item-neighborhood is built consisting of all similar items which the user has rated previously. Then that user’s rating for a different new item is predicted by calculating the weighted average of all ratings present in a similar item-neighborhood as show in the figure 3.



*Fig. 3 Item-based collaborative filtering*

As stated by Nilashi et al. (2013), all collaborative filtering methods share a capability to utilize the past ratings of users in order to predict or recommend new content that an individual user will like. The real assumption is highly based in the idea of likeness between users or between products, with the similarity being expressed as a function of agreement between past ratings or preferences.

Overall, the neighborhood-based method of collaborative filtering utilizes the existing user-item ratings to determine recommendations for new items. The user-based approach focuses on finding users with similar rating patterns, while the item-based approach concentrates on identifying similar items based on user ratings. By employing these methods, the system can provide personalized recommendations that align with user preferences.

Kunal et al. (2017) classified the main advantages of neighborhood-based methods as follow:

* **Simplicity**: Neighborhood-based methods are relatively straightforward to implement, making them accessible and easy to work with.
* **Efficiency**: This method does not require expensive training phases that need to be regularly performed in large commercial applications. This saves computational resources and time.
* **Justifiability**: Neighborhood-based methods provide concise yet comprehensive explanations for their predictions. This makes it easier for users to understand why certain recommendations are made.
* **Stability**: Despite the continuous addition of items, users, and ratings typically encountered in major e-commerce applications, neighborhood-based methods remain relatively unaffected. They maintain stability in their recommendations even as the system evolves over time.

#### 2.2.3.2 Model-Based Recommendation Methods

The model-based recommendation method differs from neighborhood-based systems in that it doesn't directly use stored ratings for prediction. Instead, it utilizes these ratings to acquire knowledge and build a predictive model. In model-based approaches, the aim is to replicate and model the interactions between users and items, incorporating factors that represent the hidden characteristics of both users and items within the system. These factors can include user preferences and item categories, among others.

To implement a model-based system, the available data (usually in the form of a training dataset) is used to train the predictive model. The model is trained to understand the patterns and relationships between users, items, and their hidden characteristics. Once the model is trained, it can be applied to predict ratings for new items based on user behavior and item attributes.

Numerous model-based approaches exist for the task of item recommendation, each employing specific techniques. These techniques include Bayesian Clustering, which utilizes probabilistic clustering algorithms; Latent Semantic Analysis, which applies matrix factorization techniques to uncover latent factors in user-item interactions; Support Vector Machines, which employ machine learning algorithms for classification and regression tasks; and Singular Value Decomposition, which decomposes the user-item rating matrix into lower-dimensional representations (Nilashi, Bagherifard, et al., 2012; Geetha & Renuka, 2019)

These model-based approaches offer a more sophisticated and comprehensive way to make recommendations by capturing underlying patterns and relationships within the data. They leverage machine learning and statistical techniques to create predictive models that can effectively estimate user ratings for new items, enhancing the recommendation process.

##### 2.2.3.2.1 Model-based collaborative filtering models

* Decision and Regression Trees: Decision trees are widely used machine learning algorithms known for their simplicity and interpretability. In collaborative filtering, decision trees are employed to hierarchically partition the data space using split criteria based on independent variables. For instance, in a binary decision tree, one branch predominantly contains one class while the other branch predominantly contains the other class. Various adaptations have been proposed to make decision and regression trees suitable for collaborative filtering recommender systems (Aggarwal, 2016).
* Naive Bayes: Naive Bayes is a simple yet remarkably powerful predictive modeling algorithm. It calculates probabilities for each factor and class (categorical value) based on instance data and selects the outcome with the highest probability. In the context of recommender systems, items are typically treated as features, and users are considered as instances to infer missing entries using a classification model. To adapt Naive Bayes to the recommender systems domain, additional measures are incorporated, particularly addressing the challenge of rating sparsity (Aggarwal, 2016; Valdiviezo-Diaz, Ortega, Cobos & Lara-Cabrera, 2019).
* Rule-based: Association rules learning was initially developed to discover patterns between products in large transaction datasets recorded by point-of-sale (POS) systems in supermarkets. For example, if a customer purchases paint and tape together, there is a likelihood they will also buy a brush. Association rules are valuable for generating recommendations when dealing with unary ratings matrices. The process involves identifying all association rules triggered by a particular customer and sorting them based on decreasing confidence. The top-k items derived from these rules are recommended to the customer. The literature on recommender systems encompasses various modifications and extensions of rule-based collaborative filtering (Aggarwal, 2016).
* Matrix factorization (MF): Matrix factorization is a widely adopted model-based collaborative filtering technique due to its accuracy, simplicity, and interpretability. The ratings given by users to items are represented using latent factors that capture the underlying features of users and items. By reducing the dimensionality of the ratings information, predictions are obtained by taking the dot product of the users' and items' hidden factors. The literature on MF encompasses numerous variations and refinements (Aggarwal, 2016; Bobadilla, Alonso & Hernando, 2020; Valdiviezo-Diaz, Ortega, Cobos & Lara-Cabrera, 2019).
* Deep learning techniques: Neural Collaborative Filtering (NCF) is one of the popular deep learning techniques used in collaborative filtering. NCF replaces the traditional inner product operation of Matrix Factorization with a neural architecture that can learn complex functions from data. It offers a generic framework that can encompass and generalize matrix factorization. Notably, NCF has gained prominence in the field of recommender systems (He, Liao, Zhang et al., 2017).

### 2.2.4 Hybrid filtering approach

Hybrid recommender systems have gained popularity as an effective approach to enhancing the accuracy of predictions in recommender systems. These systems aim to achieve better recommendation accuracy by combining collaborative filtering, content-based filtering, and other techniques. The hybrid approach can be implemented in several ways. Empirical evaluations consistently demonstrate that hybrid methods generate more accurate recommendations than independent approaches, such as pure collaborative and content-based methods (Da'u & Salim, 2019).

According to Roy and Dutta (2022), hybrid techniques involve the combination of two or more recommendation techniques to overcome limitations and enhance the accuracy and effectiveness of individual techniques. These combinations generally lead to improved performance and accuracy in recommender applications. Different approaches can be used to incorporate different techniques, such as combining the outcomes of separate techniques or employing content-based filtering within a collaborative method. Common approaches to hybridization include meta-level, feature-augmentation, feature-combination, mixed hybridization, cascade hybridization, switching hybridization, and weighted hybridization.

Geetha, Safa, Fancy, and Saranya (2018) also state that a hybrid recommender system combines multiple recommendation techniques to generate recommendations. Compared to collaborative or content-based systems, hybrid systems typically achieve higher recommendation accuracy. By combining both approaches, the collective knowledge is increased, leading to more informed recommendations. This knowledge enhancement presents promising opportunities to explore novel ways of enriching collaborative filtering algorithms with content data and content-based algorithms with user behavior data.

Netflix serves as an exemplary case of the successful implementation of hybrid recommender systems (Balush, Vysotska & Albota et al., 2021). The recommendations provided by Netflix are generated by analyzing the viewing and searching history of similar users (collaborative filtering) and suggesting movies that share similar characteristics with highly rated movies from a user's past preferences (content-based filtering) (Geetha & Renuka, 2019).

In summary, the authors agree that hybrid recommender systems, which combine collaborative filtering, content-based filtering, and other techniques, are effective in improving the accuracy of recommendations. They emphasize that hybridization enhances the performance and accuracy of individual techniques by overcoming limitations. Various approaches can be used to combine different techniques, leading to improved recommendation outcomes. The authors also highlight that hybrid systems typically achieve higher recommendation accuracy compared to purely collaborative or content-based systems. By combining approaches, the collective knowledge is increased, enabling more informed recommendations. The successful implementation of hybrid recommender systems by Netflix serves as a notable example.

#### 2.2.4.1 Types of hybrid recommendation systems

According to Çano and Morisio (2019), the different types of hybrid recommender systems can be described as follows:

* Weighted: In a weighted recommendation system, multiple recommender systems are utilized in parallel. Each model's output is combined with static weightings. For example, a content-based model and an item-item collaborative filtering model can be combined, with each contributing 50% weight to the final prediction. The advantage of the weighted hybrid approach is its ability to integrate multiple models in a linear manner, supporting the recommendation process. Weighted hybrids are commonly used due to their simplicity and flexibility (Çano & Morisio, 2019).
* Switching: The switching hybrid selects a single recommendation system based on the situation. For datasets that exhibit item-level sensitivity, the recommender selector criteria can be based on user profiles or other relevant features. This approach adds an additional layer to the recommendation model, allowing it to select the appropriate model for each situation. The switching hybrid is sensitive to the strengths and weaknesses of the constituent recommendation models (Çano & Morisio, 2019).
* Mixed: The mixed hybrid approach generates different sets of candidate datasets based on user profiles and features. The recommendation system then inputs these candidate sets into the recommendation model accordingly, combining the predictions to produce the final recommendation. The mixed hybrid system is capable of making a large number of recommendations simultaneously, leveraging the appropriate model for each partial dataset to improve performance (Çano & Morisio, 2019).
* Feature combination: In the feature combination hybrid, a virtual contributing recommendation model is introduced to the system. This model generates additional features that complement the original user profile dataset. For example, features from a collaborative recommendation model can be incorporated into a content-based recommendation model. The hybrid model considers both collaborative and content-based data from the subsystems, rather than relying solely on one model (Çano & Morisio, 2019).
* Feature augmentation: Feature augmentation hybrids involve combining two techniques, where the output of one technique is used to enhance the operation of the other recommendation technique. The second technique relies on the output of the first, making these hybrids order-sensitive. For instance, an association rules engine can generate similar items for any given item, which can then be used as augmented item attributes within a second recommender to improve its recommendations (Çano & Morisio, 2019).
* Cascade: Cascade hybrids follow a strict hierarchical structure in the recommendation system. The main recommendation system produces the primary result, and a secondary model is used to address minor issues, such as breaking ties in scoring. Successor recommendations are constrained by the predecessor, resulting in precise and controlled outcomes without introducing additional items (Çano & Morisio, 2019).
* Meta-Level: Meta-level hybrids are similar to feature augmentation hybrids, where a contributing model provides an augmented dataset to the main recommendation model. However, in meta-level hybrids, the original dataset is replaced with a learned model from the contributing model as the input to the main recommendation model. In short, the successor model leverages a model delta built by the predecessor. Unlike in cascade hybrids, subsequent recommenders in meta-level hybrids have no restrictions on the items they can recommend (Çano & Morisio, 2019).

### 2.2.5 A Comparative Analysis of Traditional Recommendation Algorithms

Over the past few years, the field of recommendation systems has experienced notable progress, fueled by the growing abundance of extensive datasets and the emergence of groundbreaking algorithms. Traditional recommendation algorithms have played a pivotal role in establishing the fundamentals of this discipline. In their study, Balush, Vysotska, and Albota (2021) conducted a comprehensive comparative analysis of these conventional recommendations. They meticulously scrutinized prominent recommendation algorithms, elucidating their inherent strengths, limitations, and specific domains of application.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm** | **Description** | **Strengths** | **Limitations** | **Applications** |
| Collaborative Filtering | Utilizes past user-item interactions to generate recommendations. | Effective with explicit feedback data. | * Cold-start problem for new users or items. * A lot of information about user ratings is needed. | E-commerce, movie/music or book recommendations. |
| Content-Based Filtering | Recommends items based on their similarity to items the user has shown interest in. | * Incorporates item characteristics and attributes. * Works instantly, even for first users. * Works correctly, even with a small amount of data. | * May lack serendipity and diverse recommendation. * Tied to the content of the service. * Not based on the wishes of users. | News articles, document recommendations. |
| Hybrid Approaches | Combines multiple algorithms to leverage their strengths and mitigate weaknesses. | Improved accuracy and flexibility. | Complexity in combining and integrating algorithms. | E-commerce, personalized recommendations. |

## 2.3 Deep Learning in Recommendation Systems

### 2.3.1 Introduction to deep learning

Deep learning, also known as deep structured learning or hierarchical learning, is a branch of machine learning that encompasses a range of techniques designed to create high-level abstract models from data. The architecture of deep learning draws inspiration from artificial intelligence and seeks to simulate the deep learning process and sensory layers of the human brain. By learning from past experiences, deep learning models can make decisions to solve specific problems encountered by machines (Bochie, Gilbert, Gantert et al., 2021).

One of the primary objectives of deep learning techniques is to automatically extract relevant features and abstractions from data. This process involves identifying the most critical information required for a given task (Mijwil & Abttan, 2021). These techniques are particularly effective when dealing with large volumes of unsupervised data, enabling the models to learn representations that capture the inherent structure and patterns present in the data. Deep learning models are capable of achieving high accuracy in tasks such as data classification and decision-making (Al-Zubaidi, Mijwil & Alsaadi, 2019).

The field of image recognition has witnessed the remarkable impact of deep learning techniques. In the 2016 ImageNet image classification competition, deep learning achieved unprecedented accuracy, surpassing 97% (Haiming et al., 2021). This breakthrough showcases the power of deep learning in surpassing traditional methods and pushing the boundaries of image recognition capabilities.

Deep learning has also revolutionized machine translation, particularly with the development of the Google Neural Machine Translation System (GNMT). Wu, Schuster, Chen, et al. (2016) demonstrated that GNMT achieved performance levels close to human translation in both English-Spanish and English-French translations. This advancement not only improves communication across language barriers but also showcases the potential of deep learning to tackle complex language tasks.

Furthermore, deep learning has made significant contributions to the field of speech recognition. Leading players in this domain, including Baidu, Xunfei, and Sogou, have utilized deep learning techniques to achieve Chinese speech recognition accuracies exceeding 97% (Haiming et al., 2021). Such breakthroughs in speech recognition have paved the way for applications in voice assistants, transcription services, and other speech-related tasks.

In the realm of online advertising, deep learning has emerged as a powerful tool for predicting click-through rates. Industry giants like Google and Microsoft have leveraged deep learning to improve the accuracy of their advertising platforms (Cheng, Koc, Harmsen, et al., 2016; Zhu, Shan, Mao, 2017). This application of deep learning has revolutionized online advertising by enabling targeted and personalized ad delivery, optimizing marketing campaigns, and enhancing user experience.

Deep learning has experienced remarkable growth in the realms of big data and artificial intelligence (Silver, Huang, Maddison, et al., 2016). By combining low-level features, deep learning creates richer, high-level abstractions that automatically uncover distributed data features. This transformative approach eliminates the need for manual feature design in traditional machine learning, leading to breakthroughs in various domains, including image recognition, machine translation, speech recognition, online advertising, and recommendation systems (Haiming, Kaili, Yunyun & Xuefeng, 2021).

In recent years, deep learning techniques have found applications in various domains, including the classification of medical images. Researchers have successfully applied deep learning models to automatically analyze and interpret medical images, aiding in the diagnosis of diseases and the identification of anomalies (Mehta, Aggarwal, Koundal et al., 2021). By leveraging the hierarchical learning capabilities of deep learning, these models can extract intricate features and patterns from medical images, enabling accurate and efficient analysis (Li, Zhao, et al., 2021).

The growth of deep learning in big data and artificial intelligence has been fueled by its ability to automatically extract meaningful features from data, surpassing traditional machine learning approaches. Its impact spans across diverse domains, including image recognition, machine translation, speech recognition, and online advertising. These advancements have not only achieved unprecedented levels of accuracy but also opened new possibilities for solving complex problems and driving innovation in various industries (Zhang, Yang, Chen & Li, 2017).

### 2.3.2 Advantages of deep learning in recommendation systems

Deep learning has revolutionized various domains, and one area where it has made significant advancements is recommendation systems. Numerous articles and journals highlight the advantages of deep learning in recommendation systems. This section discusses some of the key advantages, drawing references from relevant articles.

1. Enhanced Representation Learning:

Deep learning models, such as deep neural networks, excel at learning complex patterns and representations from raw data. This ability is particularly advantageous in recommendation systems, as it enables the extraction of intricate features from user behavior and item attributes. According to Cheng et al. (2016), deep learning techniques facilitate the automatic discovery of latent features that capture users' preferences and item characteristics, leading to improved recommendation accuracy.

1. Improved Recommendation Accuracy:

Deep learning models have demonstrated superior performance in recommendation tasks compared to traditional techniques. By employing deep neural networks, recommendation systems can effectively capture nonlinear relationships and exploit intricate dependencies between users and items. Bell and Koren (2007) highlight that deep learning algorithms, such as deep belief networks and autoencoders, have shown promising results in enhancing recommendation accuracy by effectively modeling complex user-item interactions.

1. Handling Sparse and Cold-Start Problems:

Recommendation systems often encounter sparse data, where there are limited interactions between users and items. Deep learning methods can address this challenge by effectively handling sparse data representation. Zhang et al. (2019) propose a deep collaborative filtering model that incorporates side information, such as user and item attributes, to overcome data sparsity. This approach leverages deep learning techniques to learn meaningful representations from auxiliary data, thereby improving recommendation performance in sparse scenarios.

1. Scalability and Adaptability:

Deep learning models exhibit scalability and adaptability, making them suitable for large-scale recommendation systems. They can handle massive datasets and efficiently train on powerful computational resources. Furthermore, deep learning models can adapt to evolving user preferences and item characteristics over time. Hidasi et al. (2015) propose a recurrent neural network-based recommendation model that captures temporal dynamics, enabling personalized recommendations that adapt to users' changing interests.

1. Handling Heterogeneous Data:

Recommendation systems often incorporate various data sources, including textual information, images, and social network connections. Deep learning techniques excel at handling heterogeneous data types and effectively extracting meaningful representations from diverse sources. According to Covington et al. (2016), deep learning models, such as wide and deep neural networks, can integrate multiple data inputs to capture both memorization (exploitation of past behavior) and generalization (discovering new preferences) in recommendation systems.

In summary, deep learning offers several advantages in recommendation systems, including enhanced representation learning, improved recommendation accuracy, handling sparse and cold-start problems, scalability and adaptability, and handling heterogeneous data. These advantages have been highlighted in various articles, including Cheng et al. (2016), Bell and Koren (2007), Zhang et al. (2019), Hidasi et al. (2015), and Covington et al. (2016). The integration of deep learning techniques in recommendation systems continues to advance the field, providing more accurate and personalized recommendations to users.

### 2.3.3 Deep learning architectures commonly used for recommendations

Deep learning architectures have significantly advanced the field of recommendation systems, providing improved accuracy and performance. Several deep learning models have been developed and widely used for recommendation tasks. This section discusses some of the commonly employed deep learning architectures in recommendation systems, based on relevant literature.

1. Convolutional Neural Networks (CNNs):

The Convolutional Neural Network (CNN) is a multi-layer perceptron architecture commonly utilized for processing two-dimensional image data. It consists of an input layer, convolution layer, pooling layer, fully connected layer, and output layer. Unlike traditional multi-layer perceptrons, CNNs incorporate pooling operations to reduce the number of neurons in the model and exhibit enhanced robustness to translation invariance within the input space (Zhang, Lu, Jin, 2020).

CNNs have found extensive applications in recommendation systems (Rawat & Kankanhalli, 2016). They are primarily employed to extract hidden features from various types of content, including images, text, and audio, in combination with user implicit representations (Balasubramanian, Diwan & Vora, 2020). This enables the generation of personalized recommendations for users, with typical applications in image recommendation, music recommendation, text recommendation, and more (Haiming, et al., 2021).

The strength of Convolutional Neural Networks lies in their ability to effectively capture local patterns and features, making them well-suited for modeling item content or user behavior sequences. For instance, He et al. (2017) propose a deep collaborative filtering model that leverages CNNs to capture item content information, such as images or textual descriptions, thereby enhancing the overall recommendation performance.

1. Recurrent Neural Networks (RNNs):

The use of Recurrent Neural Networks (RNNs) in recommendation systems has gained significant popularity due to their ability to capture sequential dependencies and temporal dynamics in user behavior. Unlike normal fully connected networks or convolutional neural networks, where the layers are fully connected and the nodes between each layer are disconnected, RNNs have interconnected nodes between their layers. This connectivity allows RNNs to calculate the output of the hidden layer at the current moment by considering both the input layer's output and the hidden layer's state from the previous moment (Haiming et al., 2021).

RNNs, specifically Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), have proven effective in modeling sequential patterns and capturing long-term dependencies. For instance, Hidasi et al. (2015) proposed a session-based recommendation model that leverages RNNs to capture sequential user interactions and provide personalized recommendations. By employing RNNs, this model can effectively consider the order and context of user actions to generate accurate and relevant recommendations.

1. Autoencoders:

Autoencoders are a type of unsupervised deep learning model that aims to learn efficient data representations by reconstructing input data from compressed latent representations (Wang, Yao & Zhao, 2016). In contrast to traditional neural networks, autoencoders learn the hidden layer representation of the data through an encoding and decoding process (Haiming et al., 2021). The basic structure of an autoencoder consists of three layers: an input layer, a hidden layer, and an output layer.

The primary objective of an autoencoder is to minimize the discrepancy between the input and output. However, if the model is trained solely by minimizing the error between the input and output, it may learn to perform a trivial identity function. To address this issue, researchers have proposed several variants of autoencoders, including sparse autoencoders and denoising autoencoders (Bengio, Lamblin & Popovici, 2007).

In the context of recommendation systems, autoencoders are primarily employed to learn hidden feature representations of users and items. These representations are then used to predict users' preferences for items. The application scenarios of autoencoders in recommendation systems encompass scoring prediction, text recommendation, image recommendation, and more (Haiming et al., 2021).

Autoencoders have demonstrated successful applications in recommendation systems, particularly for collaborative filtering tasks. The denoising autoencoder approach, introduced by Vincent et al. (2010), is capable of handling missing or noisy user-item interaction data. By effectively learning robust latent representations, denoising autoencoders contribute to more accurate and reliable recommendations.

1. Variational Autoencoders (VAEs):

Variational Autoencoders are generative models that combine autoencoders with probabilistic techniques, allowing the modeling of uncertainty in recommendation tasks. VAEs enable the generation of diverse recommendations and provide a continuous latent space representation for users and items. Liang et al. (2018) propose a VAE-based recommendation model that incorporates both collaborative filtering and content-based information, producing accurate and diverse recommendations.

1. Transformer-based Models:

Transformer-based models, such as the Transformer architecture introduced by Vaswani et al. (2017), have gained popularity in recommendation systems. Transformers are designed to capture long-range dependencies and enable parallel computation, making them efficient for large-scale recommendation tasks. Several transformer-based models, such as BERT and GPT, have been adapted for recommendation systems, effectively modeling user-item interactions and capturing contextual information.

In summary, deep learning architectures commonly used in recommendation systems include Convolutional Neural Networks (CNNs) for item content modeling, Recurrent Neural Networks (RNNs) for sequential and temporal modeling, Autoencoders and Variational Autoencoders (VAEs) for collaborative filtering tasks, and Transformer-based models for capturing long-range dependencies and contextual information. These architectures have been proposed and applied in various research studies, including He et al. (2017), Hidasi et al. (2015), Vincent et al. (2010), Liang et al. (2018), and Vaswani et al. (2017). Leveraging these deep learning architectures allows recommendation systems to effectively model complex user-item interactions and provide accurate and personalized recommendations.

## 2.4 Deep learning Models for Book Recommendations

Deep learning models have significantly advanced the field of book recommendation systems, enabling accurate and personalized recommendations to users. Researchers have explored various deep learning approaches to enhance book recommendations. This section provides an overview of the key methodologies and techniques, drawing references from relevant articles published from 2018 onwards.

### 2.4.1 Neural Network-Based Models

Neural network-based models have proven to be effective in book recommendation systems, leveraging their ability to capture complex patterns and representations. Wang et al. (2018) propose a book recommendation model based on a deep neural network that incorporates both textual book content and user behavior data. The model utilizes techniques such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to extract meaningful features from book descriptions and user preferences, leading to improved recommendation accuracy. This approach demonstrates the capability of neural network-based models to leverage rich book content information for better recommendations.

### 2.4.2. Collaborative Filtering with Deep Learning

Collaborative filtering, a popular technique in recommendation systems, has been enhanced by incorporating deep learning methods. Deep collaborative filtering models have shown promising results in book recommendations. Zhang et al. (2019) propose a deep collaborative filtering model that combines matrix factorization with deep neural networks. The model captures both explicit user-item interactions, such as ratings and reviews, as well as implicit interactions, such as user browsing and purchase history. By leveraging the power of deep learning, this hybrid approach significantly improves recommendation accuracy for books. It highlights the capability of deep learning models to capture intricate user-item relationships and preferences.

### 2.4.3 Hybrid Models Combining Deep Learning and Traditional Techniques

Hybrid models that combine deep learning approaches with traditional recommendation techniques have gained attention in book recommendation systems. These models aim to leverage the strengths of both approaches to provide accurate and diverse recommendations. Huang et al. (2018) propose a hybrid recommendation model that combines deep learning-based content analysis with collaborative filtering. The model utilizes deep neural networks to extract book content features and user preferences and combines them with collaborative filtering techniques to generate personalized recommendations. This hybrid approach effectively captures both the semantic meaning of books and user-item interactions, resulting in improved recommendation quality.

Furthermore, other hybrid models have been explored, such as the combination of deep learning with knowledge graph embeddings. Chen et al. (2019) propose a hybrid model that incorporates both collaborative filtering and knowledge graph embeddings to enhance book recommendations. By integrating deep learning techniques with structured knowledge about books and user preferences, this approach achieves better recommendation performance by capturing both content-based and collaborative filtering signals.

In summary, deep learning models have significantly advanced book recommendation systems. Neural network-based models effectively capture complex patterns and representations from book content and user behavior. Collaborative filtering models with deep learning techniques leverage both explicit and implicit user-item interactions. Hybrid models combining deep learning with traditional techniques capture both book content features and user preferences, leading to accurate and diverse recommendations. These approaches demonstrate the substantial progress and potential of deep learning in enhancing book recommendations.

## 2.5 Evaluation Metrics and Techniques

In order to assess the effectiveness and performance of personalized book recommendation systems, various evaluation metrics and techniques have been employed. These metrics provide insights into the accuracy, diversity, coverage, and novelty of the recommendations, enabling researchers and practitioners to measure the quality of their systems and compare them against existing approaches. This section discusses common evaluation metrics for book recommendation systems.

### 2.5.1 Common Evaluation Metrics for Book Recommendation Systems

#### 2.5.1.1 Mean Square Error

Mean Square Error (MSE) is a commonly used evaluation metric for rating prediction accuracy in book recommendation systems. It measures the average squared difference between the predicted ratings and the actual ratings for a set of recommendations (Koren et al., 2009).

MSE =

where N is the total number of recommendations, is the predicted rating for the i-th recommendation, and is the corresponding actual rating.

#### 2.5.1.2 Precision and Recall

Precision (P) measures the proportion of relevant recommendations out of the total recommendations made. It focuses on the accuracy of the system by evaluating the fraction of correctly recommended items. Precision is computed as the ratio of true positives (TP) to the sum of true positives (TP) and false positives (FP).

P =

Recall (R) measures the proportion of relevant recommendations that are successfully retrieved by the system. It focuses on the system's ability to capture all the relevant items. Recall is computed as the ratio of true positives (TP) to the sum of true positives and false negatives (FN).

R =

These metrics are particularly useful when dealing with binary relevance judgments, where a book is either relevant or not. By calculating precision and recall, researchers can determine the accuracy of the recommendations made by the system (Sarwar et al., 2018).

#### 2.5.1.3 F1-Score

F1-Score: The F1-Score is the harmonic mean of precision and recall, providing a balanced measure of the system's performance. It combines both precision and recall into a single metric, taking into account both false positives and false negatives (Zhang et al., 2018).

F1-Score

The F1-Score ranges between 0 and 1, with 1 indicating the best performance.

#### 2.5.1.4 Mean Average Precision (MAP)

Mean Average Precision (MAP) calculates the average precision at each relevant item's position and then takes the mean across all users. It considers both the relevance and the rank of recommended books. The formula for MAP is as follows:

MAP

where N is the total number of recommended items, is the position of the i-th relevant item in the recommendation list, and is the relevance of the i-th relevant item (usually represented as 1 for relevant and 0 for non-relevant) (Huang et al., 2019).

#### 2.5.1.5 Normalized Discounted Cumulative Gain (NDCG)

Normalized Discounted Cumulative Gain (NDCG) is a metric that assesses the ranking quality of recommended books. It takes into account the relevance and position of recommended items, giving higher scores to relevant books appearing at the top of the list. NDCG accounts for the diminishing returns of relevance as the position increases. The formula for NDCG is given by:

NDCG =

where DCG (Discounted Cumulative Gain) calculates the sum of relevance scores at each position, discounted by a logarithmic function, and IDCG (Ideal Discounted Cumulative Gain) represents the maximum achievable DCG. NDCG ranges from 0 to 1, with higher values indicating better performance (Li et al., 2018).

#### 2.5.1.6 Coverage and Diversity

Coverage refers to the percentage of books in the entire catalog that are recommended to users. It can be calculated using the formula:

Coverage

Diversity measures the variety of recommended books, ensuring that the system offers diverse options to cater to different user preferences. Intra-list diversity is commonly used to quantify this aspect, which calculates the average dissimilarity between pairs of recommended books in a list (Yang et al., 2018).

#### 2.5.1.7 Novelty

Novelty measures the degree to which recommended books differ from the user's prior knowledge or past interactions. Average novelty and self-information are commonly used metrics to evaluate the novelty of recommended books. Average novelty is calculated as the average dissimilarity between recommended books and the user's historical interactions. Self-information measures the information gain provided by a recommended book and is calculated using the formula:

Self-information

where P is the probability of the book being recommended (Hu et al., 2019).

In conclusion, evaluating personalized book recommendation systems requires the utilization of appropriate metrics and techniques. Precision, recall, MAP, NDCG, coverage, diversity, and novelty are commonly used metrics in this domain. These metrics provide valuable insights into the accuracy, ranking quality, coverage, diversity, and novelty of recommendations, enabling researchers and practitioners to assess and compare the performance of their systems effectively.

## 2.6 Appraisal