International Journal of Contemporary Computer Research (IJCCR), Vol.1 Issue.1 (January, 2025)
ISSN: 2600-9048

Review Paper on Recommendation Systems: Different Methods and Techniques

[1]Khlood Alrassi, [2]Yazeed Al Moaiad, [3]Khaled Alrasi

[1,2]Faculty of Computer & Information Technology, Al-Madinah International University, Kuala Lampur, Malaysia.

[3]Faculty of Information Technology, University of Zintan, Zintan, Libya.

cp516@lms.mediu.edu.my,yazeed.alsayed@mediu.edu.my, Khaled.alrasi@uoz.edu.ly

Abstract— *The rapid growth of digital content has intensified the problem of information overload, making it challenging for users to access relevant resources. Recommender systems (RSs) address this issue by filtering data and providing suggestions, thereby improving decision-making and user satisfaction. This paper presents a comprehensive review of recommender systems (RSs), with particular emphasis on their methods, techniques, benefits, history, and applications. It examines traditional approaches, including collaborative filtering, content-based filtering, and hybrid strategies, before providing a classification of deep learning models in recommender systems and analyzing their impact on enhancing RS capabilities. In addition, the paper discusses evaluation methods used to assess recommendation performance and highlights their roles in measuring system effectiveness. Finally, it synthesizes the key challenges confronting recommender systems, including data sparsity, scalability, and cold-start issues.*

Keywords — *Recommendation Systems, Traditional Methods, Evaluation Methods, Challenges.*

I. INTRODUCTION

In today's digital era, users are confronted with an overwhelming abundance of information, products, and services available through online platforms. This phenomenon, often referred to as information overload, makes it increasingly difficult for individuals to identify content that aligns with their preferences and needs. To address this challenge, recommender systems (RSs) have emerged as intelligent tools to filter large volumes of data and provide suggestions based on users' behavior, preferences, and contextual information. By reducing search effort and enhancing decision-making, RSs play a pivotal role in shaping user experiences across various domains such as e-commerce, entertainment, social media, online learning, and healthcare.

Recommender systems have significantly evolved over the past three decades. Early approaches relied primarily on traditional techniques, such as collaborative filtering, content-based filtering, and hybrid methods, which laid the groundwork for personalized recommendations. While effective, these approaches often face limitations such as data sparsity, cold-start problems, and scalability issues. With the advent of deep learning, the field has experienced transformative advancements, as neural networks and representation learning enable systems to capture complex patterns, incorporate multimodal data, and deliver more accurate and dynamic recommendations. The main objectives of this review are:

- To provide a comprehensive overview of different methods and techniques in recommender systems, examining their evolution, applications, and challenges.
- To analyze evaluation methods used in recommender systems, emphasizing their roles in assessing system performance.
- To identify the key challenges faced by recommender systems, to highlight ongoing limitations and motivate future research directions.

The key contributions of this review can be summarized as follows:

- It presents a comprehensive overview of recommendation systems, with a particular emphasis on their underlying methods and techniques.
- It provides a structured evaluation methods, including widely used protocols and performance metrics, to guide researchers in assessing system effectiveness.
- It highlights common challenges such as scalability, data sparsity, cold-start problems, explainability, and over-specialization, thereby outlining open issues for future research
- It offers a consolidated reference that bridges the theoretical framework with practical applications, serving as a resource for academic researchers.

Given these objectives and contributions, the paper is structured as follows: **Section 1** introduces the article, highlighting research objectives, contributions, and the paper's structure. **Section 2** provides the fundamental concepts, benefits, applications and history of recommender systems. **Section 3** discusses traditional methods, including collaborative, content-based, and hybrid approaches. **Section 4** explores deep learning–based recommendation models classifications, outlining their effectiveness on RSs. **Section 5** focuses on evaluation methods, discussing commonly used strategies for assessing system performance. Finally, **section 6** determines the key challenges faced by recommender systems. **Section 7** concludes the review and outlining future work.

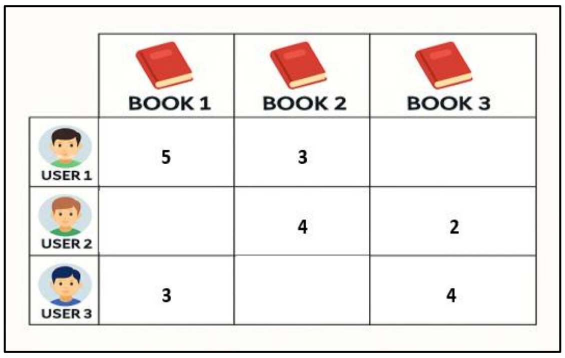
II. RECOMMENDATION SYSTEMS OVERVIEW

2.1 Definition and General Function

Recommender systems (RSs) are intelligent tools that filter large volumes of data to provide personalized suggestions based on user preferences, behaviors, and contextual information [1]. Their main goal is to reduce search effort, enhance decision-making, and improve user satisfaction by delivering ranked recommendation lists[2] [3]. These systems are widely applied across domains, such as book recommendations and e-commerce platforms like Amazon, where they tailor content and product suggestions to individual users ([1].

At the core of any recommender system lie two fundamental entities: users and items**.** Users represent individuals interacting with the system (e.g., readers, viewers, or shoppers), while items correspond to the products, services, or content available for recommendation (e.g., books, movies, or job postings) [1]. User preferences are generally captured through two feedback mechanisms: explicit ratings, where users provide direct evaluations (e.g., numerical scores or labeled intervals), and implicit ratings, which are inferred from behavioral patterns such as clicks, browsing history, or time spent on a webpage. Most modern recommender systems employ a combination of both feedback types to generate accurate and context-aware predictions.

These interactions are systematically represented within the user–item utility matrix, which constitutes the backbone of many recommendation algorithms[2, 4]. In this matrix, rows denote users and columns denote items, with each cell containing the rating or preference score a user has assigned to an item. Empty cells represent missing values, indicating that a user has not yet rated or interacted with a particular item. An example of a utility matrix is presented in Figure 1, where User 1 rated Book 1 as 5 and Book 2 as 3 but did not rate Book 3, while User 2 provided ratings for Books 2 and 3 but not Book 1. The unfilled cells highlight the predictions that algorithms, particularly collaborative filtering approaches using matrix completion, aim to estimate.



	BOOK 1	BOOK 2	BOOK 3
USER 1	5	3	
USER 2		4	2
USER 3	3		4

Figure 1: Example of a utility matrix

This utility matrix serves as the foundation for most recommendation algorithms. In collaborative filtering, the system identifies patterns among users with similar behavior to suggest items of interest, while in content-based filtering, the focus lies on comparing item features with a user's known preferences [2, 5]. This structured user–item interaction is essential for generating relevant, personalized recommendations.

To formalize this process, a recommender system is often modeled mathematically. Let U denote the set of users and I the set of all items available for recommendation. A utility function f to measure the relevance of an item i for a user u, expressed as$f : U \times I \rightarrow R$, where R is an ordered set of utility values. The system, for each user $u \in U$, recommends items $i' \in I$ that maximize the utility for this user. This formula can be expressed mathematically as in equation (1) [3]:

$$\forall u \in U,\ i'_u = \operatorname{argmax}_{i \in I} f(u, i). \qquad (1)$$

Ultimately, recommender systems can be broadly categorised into two main approaches: personalised and non-personalised. Personalized systems leverage the unique history and behavior of each user to deliver tailored recommendations, while non-personalized systems provide generalized suggestions without considering individual preferences [3].

2.2 Data Sources

Recommender systems (RSs) are computational models that generate personalized suggestions by leveraging diverse data sources, including user profiles, item attributes, and user–item interactions. The type and granularity of data required depend on the underlying approach: while some algorithms operate effectively on basic rating information, others necessitate richer and more complex datasets to uncover deeper patterns and insights [6]. These data sources are commonly categorized into three groups: user data, item data, and interaction data, each contributing uniquely to the design and effectiveness of recommendation systems.

1. User Data

Users are the recipients of recommendations, and each individual typically exhibits unique goals and preferences. To generate accurate suggestions, recommender systems collect and analyze user-specific information, the nature of which varies depending on the underlying algorithm. Traditional collaborative filtering approaches often define a user profile solely through rating histories, whereas more advanced systems incorporate socio-demographic attributes (e.g., age, gender, education, occupation) to improve personalization. These features are typically encoded into user models, often represented as vectors that capture individual tastes. Beyond static attributes, behavioral signals—such as browsing history, interaction patterns, or clickstream data—are particularly valuable, as they enable systems to adapt dynamically to evolving user interests.

2. Item Data

Items constitute the central entities of recommendation, and their representation plays a crucial role in shaping system outputs. Each item is evaluated along two key dimensions: utility, which measures its perceived relevance or benefit to the user, and information complexity, which reflects the cognitive effort required for evaluation. For instance, news articles are considered low in complexity due to concise headlines, whereas job postings are high in complexity, requiring detailed review. To facilitate effective comparison and ranking, recommender systems utilize a range of item attributes, such as ID, title, genre, or language. The integration of these attributes, along with contextual factors, directly influences how items are matched with user preferences in different recommendation scenarios.

3. Interactions data

Interaction data captures the exchanges between users and items, and serves as a key resource for inferring preferences. These data may be **explicit**, such as user-provided ratings, tags, or reviews, or **implicit**, such as clicks, search queries, or time spent engaging with content. Modern systems often integrate multiple interaction types to support diverse tasks and improve recommendation accuracy. For example, an e-commerce platform may use aggregate interaction logs to display trending products on the homepage, while leveraging fine-grained behavioral signals, such as item views or clicks, to personalize recommendations during browsing. By systematically recording and analyzing such transactions, recommender systems can refine their models and deliver context-aware, user-centric suggestions.

2.3 The Process of a Recommender System (RS)

As illustrated in Figure 2, a recommender system operates as a cyclical and structured process aimed at predicting and satisfying user needs through personalized suggestions. This workflow highlights the iterative nature of RSs, where user interactions generate data that is continuously fed back into the system for improved personalization.

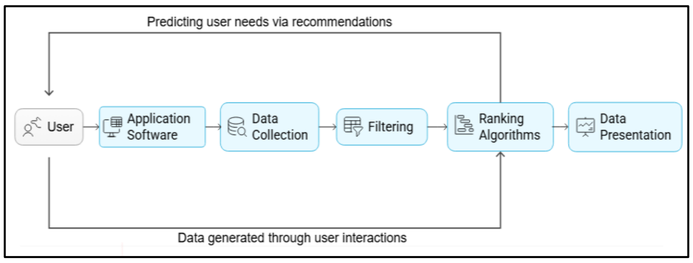


Figure 2: A workflow diagram of a recommender system process. *Source: [3].*

This process generally comprises five essential phases:

1. User Interaction Phase

Users engage with the system through an application interface, performing actions such as clicks, ratings, reviews, and purchases. These interactions provide raw behavioral signals, which are passed on for data collection and analysis.

2. Data Collection Phase

The system gathers and stores detailed user information to construct profiles that summarize individual characteristics and behaviors. Accurate profiling is crucial for effective personalization. User data is typically collected from three feedback sources. [4]:

- Explicit feedback, where users directly provide evaluations (e.g., ratings, reviews).
- Implicit feedback, inferred from behavioral patterns (e.g., clicks, browsing duration, purchase history).
- Hybrid feedback, which integrates both explicit and implicit signals to improve robustness.

This collected information serves as the foundational input for recommendation generation.

3. Filtering Phase

The aggregated data is processed using one or more filtering techniques:

- Collaborative filtering, which identifies patterns of similarity across users or items.
- Content-based filtering, which relies on item features to align with user preferences.
- Hybrid filtering, which combines multiple techniques to overcome the limitations of individual approaches and improve accuracy.

4. Ranking Algorithms Phase

The filtered items are then prioritized using ranking algorithms, which assess the relative relevance of each item to a given user's profile. This step ensures that the most suitable options appear at the top of the recommendation list.

5. Data Presentation Phase

Finally, the ranked recommendations are delivered to users through a clear and user-friendly interface, enabling seamless decision-making. This presentation closes the cycle by feeding user reactions back into the system for further refinement.

2.4 History of Recommendation Systems

The history of recommender systems reflects a trajectory of continuous innovation shaped by advances in algorithms, computational power, and industrial adoption. This evolution can be broadly divided into five major phases, each representing a distinctive shift in how recommendations are generated and delivered to users.:

1. **Early Era: Manual and Rule-Based Filtering (1992 – mid 1990s)**

The concept of recommender systems originated in the early 1990s, when Belkin and Croft [7] distinguished between information filtering and information retrieval, laying the foundation for recommender technology. That same year, Goldberg, et al. [8] introduced the first information filtering model, Tapestry, which used collaborative filtering supported by user evaluations. Soon after, systems like GroupLens introduced automated user-user collaborative filtering models, enabling scalable recommendation capabilities for larger audiences. Ringo and Video Recommender systems further extended these techniques to the music and video domains[5, 9].

2. **Growth of Collaborative Filtering and E-Commerce (Late 1990s - early 2000s)**

By the late 1990s, the commercial value of recommendation systems became evident, with companies like Net Perceptions offering personalized marketing engines for e-commerce platforms such as Amazon and Best Buy. The MovieLens project, also by GroupLens, introduced benchmark datasets that remain widely used in academic research [9].

3. **Hybridization and Matrix Factorization (mid 2000s - early 2010s)**

As simple collaborative filtering faced issues like cold-start problems, hybrid recommender systems emerged, combining collaborative, content-based, and knowledge-based methods.[10] During the Netflix Prize (2006-2009), matrix factorization techniques gained prominence for their ability to handle large-scale and sparse data. This era also saw the introduction of Factorization Machines (FM) and Field-aware Factorization Machines (FFM), which improved predictive power by modeling complex feature interactions. In 2007, the first ACM RecSys Conference was held, institutionalizing the research community around recommender systems. Today, ACM RecSys stands as one of the leading annual academic conferences dedicated to research in recommender systems [9].

4. **Deep Learning and Neural Approaches (2016 - present)**

The introduction of deep neural networks brought transformative changes. Models like YouTubeDNN, Wide & Deep, DeepFM, and Neural Collaborative Filtering (NeuralCF) enhanced the ability to model complex user-item relationships [9].Transformer-based models (e.g., BERT4Rec) and reinforcement learning frameworks (e.g., DRN, NICF) further expanded capabilities, making recommendations more context-aware and dynamic [9].

5. **Recent Trends: Graph-based and Causal Recommendation (2020s - future)**

In recent years, graph neural networks (e.g., KGAT, RippleNet) have captured relational information, improving recommendation diversity and explainability[9]. Furthermore, causal inference methods have emerged to address biases and fairness issues, signalling a shift towards more ethical and robust recommendation strategies [9].

Figure 3 illustrates the key phases in the evolution of recommender systems, from early manual filtering and collaborative methods to hybrid models, deep learning, and recent graph-based and causal approaches.

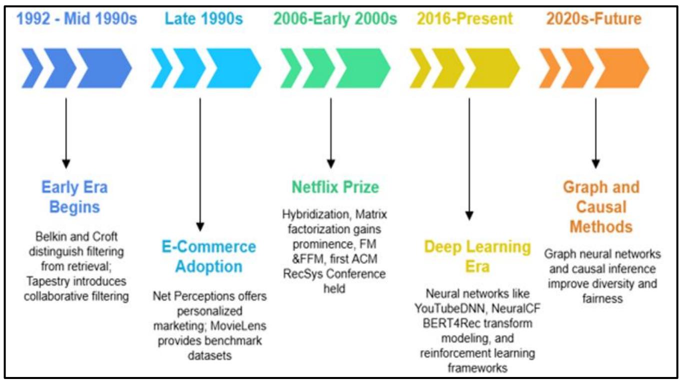

1992 - Mid 1990s
Late 1990s
2006-Early 2000s
2016-Present
2020s-Future
Early Era Begins
Belkin and Croft distinguish filtering from retrieval; Tapestry introduces collaborative filtering
E-Commerce Adoption
Net Perceptions offers personalized marketing; MovieLens provides benchmark datasets
Netflix Prize
Hybridization, Matrix factorization gains prominence, FM &FFM, first ACM RecSys Conference held
Deep Learning Era
Neural networks like YouTubeDNN, NeuralCF BERT4Rec transform modeling, and reinforcement learning frameworks
Graph and Causal Methods
Graph neural networks and causal inference improve diversity and fairness

Figure 3: The evolution of recommender systems.

2.5 Benefits of Recommender Systems (RS)

Recommender systems (RSs) provide substantial advantages to both users and service providers across a wide range of domains. As digital environments continue to expand with overwhelming volumes of content, products, and services, the capacity to deliver relevant and personalized suggestions has become increasingly critical. The benefits of RSs can be broadly examined from two perspectives: those of users and service providers [1, 6]:

2.5.1 User Benefits

From the user's perspective, recommender systems enhance decision-making, reduce information overload, and enrich the overall digital experience. The key benefits include:

- **Finding Items of Interest**

RSs help users identify content or products aligned with their preferences by providing ranked lists of suggested items. This is particularly valuable when users face difficulties locating desired content. Even when users know what they want, being exposed to related items can reinforce existing interests or inspire new ones, thereby enhancing both efficiency and discovery.

- **Discovering Groups of Related Items**

Beyond individual suggestions, RSs often provide bundle or sequential recommendations. For example, a user purchasing a soccer ball might also be presented with complementary items such as shoes or shirts. Similarly, sequential recommendations, such as playlists in music services, reduce decision fatigue by presenting logically ordered items, thereby streamlining consumption.

- **Exploring Content**

Many users interact with RSs in an exploratory manner, browsing without a specific target. Platforms such as YouTube and Pinterest exemplify this functionality by showcasing personalized and novel suggestions on their homepages. This exploratory capability fosters engagement, stimulates curiosity, and builds long-term user loyalty.

- **Improving Recommendations Over Time**

RSs continuously refine their accuracy as they learn from user interactions. With repeated use, systems capture behavioral patterns and adapt to evolving interests, providing increasingly relevant and context-aware suggestions. This iterative improvement not only benefits users but also enhances the system's predictive intelligence.

- **Assisting Others**

Some users contribute data, such as ratings or reviews, with the intention of supporting other users rather than for personal benefit. For example, in an automobile recommendation platform, a user may provide a rating after purchasing a car to help future buyers make informed decisions. This collective contribution strengthens the value of the RS ecosystem.

2.5.2 Service Provider Benefits

From the perspective of service providers, RSs deliver several strategic advantages that improve business performance and operational decision-making. [1, 6] Key benefits include:

- **Increasing Item Sales**

Commercial RSs are designed to boost sales by encouraging users to purchase additional or related items that they might not otherwise consider. In non-commercial contexts, such as news or streaming platforms, the goal shifts toward increasing content consumption. Ultimately, RSs aim to improve the *conversion rate*, ensuring that users not only browse but also act on recommendations.

- **Increasing Sales of Diverse Products**

RSs expand visibility beyond popular items by promoting less obvious or niche options to suitable users. For instance, tourism platforms can highlight lesser-known attractions tailored to individual preferences, thereby diversifying exposure and increasing overall consumption.

- **Enhancing User Satisfaction**

A well-designed RS improves the user experience by delivering accurate, relevant, and engaging suggestions. When paired with a user-friendly interface, these recommendations positively shape perceptions of the platform, leading to greater acceptance and sustained usage.

- **Increasing User Loyalty**

Personalization fosters stronger user loyalty by recognizing returning visitors and adapting recommendations based on historical activity. As users engage repeatedly, the system refines its understanding of their preferences, strengthening their attachment to the platform and encouraging long-term retention.

- **Gaining Deeper Insights into User Preferences**

RSs capture and represent user preferences, both explicitly (e.g., ratings, reviews) and implicitly (e.g., clickstreams, browsing history). Service providers can repurpose this information to inform decision-making in areas such as inventory management, targeted marketing, or content development. For example, in tourism, destination management organizations may use interaction data to tailor promotional strategies or identify new customer segments.

2.6 Recommendation System Applications

With the exponential growth of data and the rapid expansion of online services, users are increasingly confronted with an overwhelming number of choices. In this context, recommender systems (RSs) have become indispensable tools for navigating vast information spaces and supporting informed decision-making. By delivering personalized and context-aware suggestions, RSs enhance user experiences, reduce search effort, and alleviate cognitive load. Their growing importance can be examined across three dimensions: industry adoption, academic research, and the availability of tools and commercial solutions.

- **Industry Adoption**

Major technology companies have integrated recommender systems as central components of their services. Examples include Amazon (e-commerce), YouTube (video streaming), Facebook (social media), LinkedIn (professional networking), Spotify (music streaming), Last.fm (online radio), IMDb (movie reviews), and TripAdvisor (travel advisory). These platforms rely on RSs to boost engagement, personalize user experiences, and increase retention. A notable milestone was the *Netflix Prize*, which offered a $1 million reward for improving the company's recommendation algorithm, highlighting the commercial significance of RS innovation [6].

- **Academic Research and Education**

RSs have also become a vibrant research field within academia. Dedicated conferences, such as the ACM Conference on Recommender Systems (RecSys), serve as premier venues for presenting cutting-edge work, while broader conferences, including the Web Conference (WWW), the Conference on Knowledge Discovery and Data Mining (KDD), and the SIGIR Conference, consistently feature RS-related research. In addition, the proliferation of university courses, textbooks, and journal publications reflects the growing academic attention and educational emphasis on this domain.[6].

- **Availability of Tools and Commercial Solutions**

The accessibility of RS technologies has expanded considerably. Open-source implementations are widely available through platforms such as GitHub, with major contributors like Microsoft releasing freely usable frameworks. Furthermore, commercial cloud-based solutions, such as *Amazon Web Services (AWS) Personalize* and *Google Recommendations AI*, provide scalable and customizable recommendation services. These offerings democratize access, enabling not only large enterprises and researchers but also smaller businesses and individual developers to deploy advanced recommendation capabilities.

As a result of these developments, RSs have become integral to numerous domains, ranging from entertainment and education to e-commerce and social networking. Examples of application areas are summarized in Table 1.

Table 1
Examples of popular platforms and their recommended content.

Platform Domain	Examples	Type of Recommendations
Entertainment	Netflix, Spotify, YouTube	Movies, songs, and shows based on user preferences
Information Content Platforms	Google News, Bing News, and E-learning platforms	News articles, web pages, and educational content
Online Services	Travel sites, Matchmaking services, Professional networks	Travel destinations, consultants, and matchmaking suggestions.
E-commerce	Amazon	Products, complementary or sequential items
Social Platforms	Facebook, LinkedIn, Twitter	Friends, groups, posts, and advertisements.
Other Domains	Customer support, Insurance services, Conversational agents	Solutions, policies, support content, chatbot replies

III. Traditional Recommendation Systems Methods

Recommender system development has been shaped by a variety of methodological approaches, each of which is designed to address the challenge of filtering vast amounts of information and delivering suggestions. Traditional methods laid the groundwork for modern recommendation technologies and remain central in both academic research and industrial applications. These methods differ in how they model user preferences, represent item features, and generate predictions, yet they share the common goal of enhancing user experience by reducing information overload.

Recommendation system (RS) methods have historically been classified into several categories, including collaborative filtering, content-based filtering, utility-based methods, demographic-based methods, knowledge-based approaches, and hybrid methods [11]. Each of these approaches employs distinct principles and techniques to generate recommendations, with their applicability varying according to data characteristics and domain requirements. Among these, collaborative filtering, content-based filtering, and hybrid approaches have emerged as the most widely adopted due to their effectiveness and adaptability across diverse application areas.

3.1 Collaborative Filtering (CF)

Collaborative Filtering (CF) represents one of the earliest and most fundamental methodologies in the evolution of recommendation systems. The concept of CF was first introduced by Goldberg et al. in 1992 [5]. Since then, it has since become an essential method in personalized recommendation technologies. This method generates personalized recommendations for users based on their preferences and the behaviors of similar users. The underlying premise of this approach is that users with similar preferences are likely to exhibit comparable interests and, consequently, enjoy similar items. By leveraging the collective behaviors and preferences of a user community, CF can generate tailored recommendations that align closely with individual tastes (Wu, 2022). Figure 4 illustrates a collaborative filtering recommendation system.

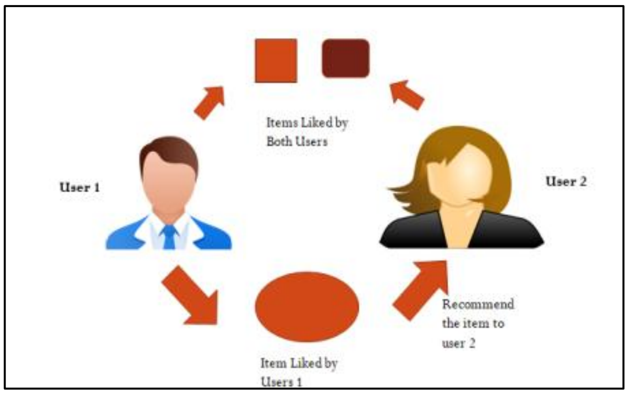


Figure 4: collaborative filtering.
Source :[12]

Collaborative Filtering Classifications

Collaborative filtering generally can be classified into two key categories: memory-based methods and model-based methods[13].

3.1.1.1 Memory-Based Collaborative Filtering

Memory-based CF methods generate recommendations by considering the preferences of neighboring users. These approaches rely directly on the utility matrix to make predictions. The model can be formally represented as a function, where the utility matrix serves as the input [2].

- **Mathematical Representation of the Memory-Based CF**

The memory-based collaborative filtering model can be mathematically formulated as shown in equation (2):

Model = f (Utility Matrix). (2)

In this representation, *f* denotes the function or algorithm that processes the input, which is the utility matrix containing user-item interactions. The output of this function is a set of predicted ratings or preferences for previously unrated items. These predictions are subsequently used to generate personalized recommendations for each user.[2].

Building upon this foundation, once the model has been established, the recommendation process advances through a function that takes both the defined model and the user's profile as inputs. However, this approach is inherently restricted to users whose profiles are already present in the utility matrix. To extend recommendations to a new user, their profile must first be incorporated into the matrix. This integration requires recalculating the similarity matrix to reflect the updated user data, thereby enabling the system to identify relevant neighbors for the new user. As a result, this process becomes computationally demanding, especially in dynamic environments characterized by frequent user additions and updates.

- Recommendation Process of the Memory-Based CF

Following the construction of the model, recommendations are produced using a function that considers both the defined model and the user's profile as inputs. This relationship can be expressed as presented in the formula (3) [2]:

Recommendation = f (Defined Model, User Profile) . (3)

Where, *User Profile* $\in$ *Utility Matrix.*

This formulation indicates that recommendations can only be generated for users whose profiles are present within the utility matrix. To accommodate a new user, their profile must first be incorporated into the matrix. This integration requires the system to recompute the similarity matrix to reflect the updated user-item relationships. Consequently, this step increases computational demands, particularly in large-scale or frequently updated recommendation systems.

In addition, Memory-Based Collaborative Filtering employs techniques such as Pearson Correlation, Vector Cosine Similarity, and K-Nearest Neighbors (KNN) to identify similar user groups, or neighborhoods, and subsequently recommend items to users within these groups. Furthermore, this method is classified into two types: user-based collaborative filtering and item-based collaborative filtering[13]. User-based collaborative filtering functions by identifying similarities among users through the comparison of their ratings on shared items. Based on ratings provided by similar users, the model the model generates a ranked list of the top N items that best align with the target user's preferences [2, 13]. In contrast, item-based collaborative filtering estimates a user's preference for a given item by evaluating its similarity to items the user has previously interacted with, as derived from the user-item rating matrix[14]. Figure 5 illustrates the difference between user-based and item-based collaborative filtering.

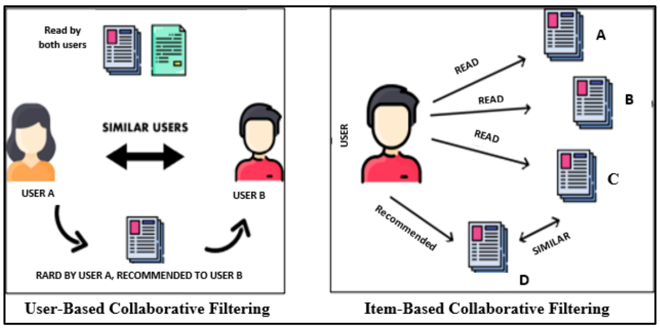

Read by both users
SIMILAR USERS
USER A
USER B
RARD BY USER A, RECOMMENDED TO USER B
User-Based Collaborative Filtering
USER
READ
READ
READ
A
B
C
Recommended
SIMILAR
D
Item-Based Collaborative Filtering

Figure 5: The processes of user-based and item-based CF.
Source :[15].

3.1.1.2 Model-Based Collaborative Filtering

Model-based collaborative filtering employs data mining and machine learning algorithms to construct predictive models that estimate a user's rating for previously unrated items. The model-based collaborative filtering methods utilize various data mining and machine learning algorithms to construct predictive models capable of estimating a user's rating for unrated items. Unlike memory-based approaches, they do not require the entire dataset during the recommendation phase. Instead, they extract significant features from the dataset to train the model, which gives rise to the term "model-based technique." The prediction process typically involves two stages: first, building the model, and second, using a function *f* that takes the defined model and the user profile as inputs to generate rating predictions, as shown in equation (3):

Recommendation = f (Defined Model, User Profile). (3)

Where, User Profile $\notin$ Utility Matrix.

A key advantage of model-based techniques is that they do not require new user profiles to be integrated into the utility matrix before making predictions. Recommendations can be provided even to users who are not part of the existing model, making these systems highly suitable for generating group recommendations. By utilizing a pre-trained model, they can quickly suggest a range of relevant items. Moreover, the accuracy of model-based systems heavily depends on the performance of the underlying learning algorithms. These methods are particularly effective in addressing common challenges in recommender systems, such as sparsity and scalability, by using dimensionality reduction and advanced model learning techniques.

3.1.2 Techniques Commonly Used in CF

CF utilized many techniques to compute the similarity between users. These techniques are mainly categorized into two types: techniques used in memory-based CF and model-based CF.

- **Techniques used in Memory-Based CF**

Typically, Memory-Based Collaborative Filtering is categorized into two primary approaches: user-based collaborative filtering and item-based collaborative filtering. Common techniques used in both user-based and item-based CF include:

1. Similarity Measures

In Collaborative Filtering, similarity measures calculate how closely two users' preferences align based on the ratings they have given to the same items. A similarity score ranges between 0 and 1, and sometimes between -1 and 1 depending on the method.

- A value closer to 1 → means the users have very similar tastes.
- A value closer to 0 → means the users have different tastes

Common similarity calculation methods involve:

a) Euclidean Distance

Euclidean distance, represented as d, measures the straight-line distance between two users u and v (or two items i and j) in a Euclidean space. Each user is treated as a point defined by their coordinates based on item ratings, and similarly, each item is defined by user ratings. The distance between two users or items is calculated as the absolute value of the difference between their respective coordinates [16]. Subsequently, the location distance between two users, represented by vectors u and v (or two items i and j), indicates how similar they are: the smaller the distance, the more similar they are. This relationship is quantified using the formula shown in the Equation: two users or items, represented by vectors u and v, indicate how similar they are; the smaller the distance, the more similar they are. [11, 17]. The Euclidean distance formula, which quantifies the similarity between users u and v, is given as shown in equation (4) [16]:

$$d(u,v) = \sqrt{\sum_{i \in I_{u\,v}} (r_{vi} - r_{ui})^2} \quad . \qquad (4)$$

In this equation, $I_{u\,v}$ represents the set of items that both users u and v have rated. The values r_{vi} and r_{ui} correspond to the ratings given by users u and v, respectively, for item i. Formula (5) calculates the Euclidean distance between two items, i and j [16]:

$$d(i,j) = \sqrt{\sum_{i \in U_{ij}} \left(r_{uj} - r_{ui}\right)^2} \quad . \qquad (5)$$

Here, U_{ij} refers to the group of users who have rated both items i and j. The terms r_{ui} and r_{uj} indicate the ratings given by user u for item i and j, respectively. To use Euclidean distance as a similarity metric, it must be normalized. Formulas (6) and (7) present the Euclidean similarity (ES) calculations for users and items, respectively [16].

$$\boldsymbol{ES(u,v)} = \frac{\mathbf{1}}{\mathbf{1+d(u,v)}} \quad . \qquad (6)$$

$$\boldsymbol{ES(i,j)} = \frac{\mathbf{1}}{\mathbf{1+d(i,j)}} \quad . \qquad (7)$$

b) Pearson correlation

Pearson correlation measures the degree of linear relationship between two users based on their ratings, producing a value ranging from -1 to 1[11]. A value of 1 signifies a strong positive relationship, (–1) indicates a strong negative relationship, and (0) means there is no correlation between the variables [18] . The similarity among two users u and v is expressed as presented in equation (8) [16] :

$$PCC(u,v) = \frac{\sum_{i \in I_{u\,v}} (r_{ui} - r^{-}{}_{u})((r_{vi} - r^{-}{}_{v})}{\sqrt{\sum_{i \in I_{u\,v}} (r_{u\,i} - r^{-}{}_{u})^2} \sqrt{\sum_{i \in I_{u\,v}} (r_{vi} - r^{-}{}_{v})^2}} \ .(8)$$

In this context, $I_{u\,v}$ represents the set of items that have been rated by both users *u* and *v*. The terms $r^{-}{}_{u}$ and $r^{-}{}_{v}$ refer to the average ratings given by users *u* and *v* on the items within $I_{u\,v}$ respectively.r_{ui} and r_{vi} are the individual ratings that users *u* and *v* assigned to the same item *i*. Formula (9) is used to compute the similarity between two items *i* and *j* [16]:

$$PCC(i,j) = \frac{\sum_{i \in U_{ij}} (r_{ui} - r^{-}{}_{i})((r_{uj} - r^{-}{}_{j})}{\sqrt{\sum_{i \in U_{ij}} (r_{u\,i} - r^{-}{}_{i})^2} \sqrt{\sum_{i \in U_{ij}} \left(r_{uj} - r^{-}{}_{j}\right)^2}} \quad . \qquad (9)$$

Here, U_{ij} refers to the group of users who have rated both items *i* and *j*. The symbols $r^{-}{}_{i}$ and $r^{-}{}_{j}$ represent the average ratings for items *i* and *j* among these users. r_{ui} and r_{uj} indicate the ratings given by user *u* to items *i* and *j*, respectively.

c) Vector Cosine Similarity

In this method, a user is represented as a vector containing their item ratings, while an item is represented as a vector of ratings given by multiple users. The cosine of the angle between two such vectors—whether for users or items—reflects their level of similarity. A cosine value close to 1 indicates a strong similarity, whereas a value near 0 suggests that the variables are unrelated or independent. Equations (10) and (11) illustrate how cosine similarity is calculated for user-based and item-based comparisons, respectively[16]:

$$Cosine\ (u,v) = \frac{\sum_{i \in I_{u\,v}} r_{ui} r_{vi}}{\sqrt{\sum_{u \in I_u} r_{ui}^2} \sqrt{\sum_{u \in I_v} r_{vi}^2}} \quad . \qquad (10)$$

In this formula, I_u and I_v represent the sets of items rated by users *u* and *v*, while $I_{u\,v}$ refers to the set of items that both users have rated. The terms r_{ui} and r_{vi} indicate the rating scores that users *u* and *v* assigned to item *i*, respectively[16].

$$Cosine\ (i,j) = \frac{\sum_{u\in U_{i\,j}} r_{ui} r_{uj}}{\sqrt{\sum_{u\in U_i} r_{ui}^2}\sqrt{\sum_{u\in U_j} r_{uj}^2}}\,. \qquad (11)$$

Here, U_i and U_j refer to the sets of users who have rated items *i* and *j*, respectively, while $U_{i\,j}$ indicates the group of users who have rated both items. The values $\boldsymbol{r_{ui}}$ and $\boldsymbol{r_{uj}}$ represent the ratings given by the same user *u* to items *i* and *j*, respectively.

d) Jaccard Index

The Jaccard index, symbolized as J, is used to evaluate the similarity and diversity between two sets. It is calculated by dividing the number of elements common to both sets (their intersection) by the total number of unique elements across both sets (their union). In other words, it reflects the proportion of shared elements relative to the total elements in the two sets. The Jaccard index ranges from 0 to 1, where values closer to 1 indicate greater similarity. Equation (12) shows how to compute the Jaccard index for two vectors u and v, which may represent either users (as sets of rated items) or items (as sets of user ratings) [16].

$$J(u,v) = \frac{|u\cap v|}{|u\cup v|}\ . \qquad (12)$$

2. Top-N Neighborhood Selection

Once similarity scores are calculated, the system selects the Top-N (or Top-K) most similar users or items to generate recommendations. One commonly used approach is the K-Nearest Neighbors (KNN) method, which functions either by locating users with comparable preferences (UserKNN) or by identifying items similar to those a user has previously engaged with (ItemKNN). These methods were originally developed to process explicit feedback, such as user rating data [19].

- **Techniques of Model-Based CF**

1. Matrix Factorization (MF)

Matrix Factorization emerged as a prominent recommendation technique following its successful application in the Netflix Prize competition, where it demonstrated a strong capability to address the issue of data sparsity commonly encountered in collaborative filtering approaches [13].This approach functions by deriving latent factors from user–item interaction data and mapping both users and items as vectors within a common latent feature space. The fundamental objective of Matrix Factorization is to identify underlying dimensions that encapsulate user preferences and characteristics by structuring the evaluation data within a rating matrix framework. Furthermore, this approach offers notable scalability and flexibility, as it can effectively incorporate both explicit feedback (e.g., user ratings) and implicit behavioral signals (e.g., search patterns and mouse movements), thereby enabling a more comprehensive analysis of user interests [13].

The two primary Matrix Factorization algorithms, Singular Value Decomposition (SVD) and Alternating Least Squares (ALS), along with their improved variants, have been widely adopted to enhance the effectiveness of recommender systems.

- Singular Value Decomposition (SVD) is a matrix factorization technique that identifies underlying features in the dataset by decomposing the original user-item rating matrix into a product of three smaller matrices[20].
- Alternating Least Squares (ALS) is a matrix factorization algorithm that d

composes the original user-item rating matrix into two separate matrices: a user-to-feature matrix and an item-to-feature matrix. It is particularly effective for handling sparse data by initially filling missing entries with random values and then iteratively minimizing the error term. This back-and-forth optimization continues until the product of the two matrices closely approximates the original user-item matrix, thereby addressing the sparsity issue[20].

3. Clustering

Clustering is a technique employed to categorize data into a limited set of groups or clusters, and it is widely utilized in recommendation systems, especially those based on Collaborative Filtering. Among the available clustering techniques, K-means is the most frequently used. This algorithm operates by first determining the desired number of clusters (K) and then organizing data points according to their closeness to the cluster centroids. Through an iterative process, each data point is assigned to the nearest cluster based on similarity metrics, and the centroids are subsequently updated[13].

3.1.3 The Process of Collaborative Filtering (CF)

The collaborative filtering technique operates through a structured and systematic process, aiming to generate recommendations based on the collective preferences and behaviors of users. This process generally comprises four essential stages:

1. Build The Utility Matrix

In this phase, user ratings or preferences, whether collected explicitly or implicitly, are gathered to construct a user-item matrix (also known as a utility matrix), as illustrated in the preceding section. Within this matrix, numerous entries are typically missing, representing the unknown user preferences that the model aims to predict.

2. Computing Similarity Between Users or Items

The system analyzes the matrix to calculate similarity scores to determine which users have comparable tastes and behaviors, and which items are alike. This analysis enables the identification of the most relevant neighbors for each user. Using these relationships, the system predicts the missing values through a process known as matrix completion, thereby estimating users' unknown preferences. To identify similarities between users, the system employs two main approaches, user-based and item-based. As explained previously, user-based collaborative filtering recommends items by identifying users with similar preferences and suggesting items that similar users have liked. Whereas item-based collaborative filtering recommends books like those a user has previously enjoyed.

3. Predict Missing Ratings (Recommendation Scores)

Once similarities are computed, missing ratings are estimated as follows:

- For User-Based Collaborative Filtering (CF)

Various methods have been introduced to estimate the rating for an active user. Among them, the weighted sum approach, as proposed by Sarwar et al. (2001), remains one of the most widely adopted techniques. This method is mathematically expressed in equation (13) [16]:

$$\hat{r}_{u\,i} = \frac{\sum v \in N_u^i \, Sim_{u\,v} \times r_{v\,i}}{\sum v \in N_u^i \, |Sim_{u\,v}|} \quad . \qquad (13)$$

Where, $\hat{r}_{u\,i}$ predicted rating of item *i* by user *v*. The set N_u^i consists of the top-k most similar users to user u who have already rated item i. The variable v represents a user within this neighborhood. The term $Sim_{u\,v}$ measures the similarity between user u and user *v*, typically based on their historical rating patterns. $r_{v\,i}$ is the actual rating that neighboring user *v* has given to item *i*.

- For Item-Based Collaborative Filtering (CF)

A variety of methods have been introduced for rating prediction. The most widely recognized approaches include Z-score normalization, the Weighted Sum method, and Mean-Centred Aggregation. The formula for the weighted sum approach is presented in equation (14) [16]:

$$\hat{r}_{u\,i} = \frac{\sum j \in N_i^u Sim_{i\,j} \times r_{uj}}{\sum j \in N_i^{iu} |Sim_{ij}|} \quad . \qquad (14)$$

In this equation, $\hat{r}_{u\,i}$ predicted rating of item i by user u. The variable j refers to an item within the neighbourhood of item i, specifically those items that are considered similar to i and have been rated by the same user. The term $Sim_{i\,j}\rightarrow$ denotes the similarity score between items i and j, which is typically calculated based on item features or user rating patterns. r_{uj} indicates the actual, known rating that user u has assigned to item j. The set N_i^u represents the neighborhood of item i for user u, which includes the top-k most similar items to i that user u has previously rated.

4. Generate Recommendations

After predicting the missing ratings, the system ranks items with the highest predicted ratings and recommends the Top-ranked items to the user. In this case, CF leverages the collective behaviors of users to suggest items that align with individual interests. These suggestions form the basis for generating recommendations.

Figure 6 illustrates the collaborative filtering process for generating item recommendations to users.

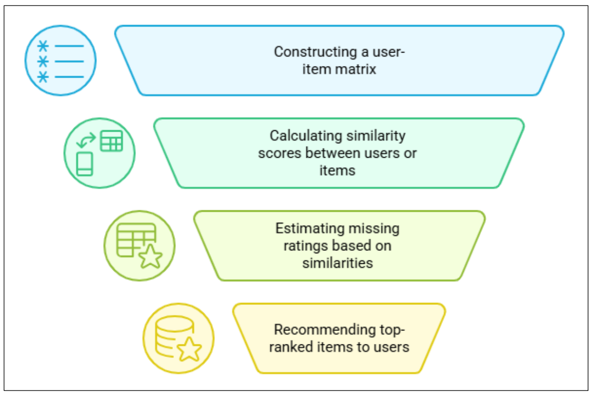

Constructing a user-item matrix
Calculating similarity scores between users or items
Estimating missing ratings based on similarities
Recommending top-ranked items to users

Figure 6: The collaborative filtering stages.

3.2 Content-Based Filtering (CBF)

Content-based filtering, a recommendation method that emerged in the early 1990s, relies on analyzing item attributes to generate recommendations. The development of this approach was influenced by the study of Loeb and Terry [21], which initiated the exploration of various models for information filtering and laid the groundwork for subsequent advancements in the field [13]. Content-based filtering suggests items by comparing the features of products a user has previously liked with those of other available items [2, 13]. This approach analyzes the content of items that a user has interacted with and recommends similar ones accordingly [3]. As a result, it requires a detailed understanding of item attributes, such as genre, author, and keywords, to produce meaningful suggestions [2].

In content-based filtering, recommendations are exclusively based on the user's data and previous activities. A core element of CBF strategy involves building a user profile that captures individual preferences derived from the descriptions of preferred items [22]. This method is commonly employed when recommending documents such as web pages, news articles, research papers, books, or restaurants. It works by comparing the attributes of potential items against the user profile. Only items that exhibit high similarity to the user's preferences are recommended, ensuring personalized and relevant suggestions[23]. Figure 7 illustrates content-based filtering:

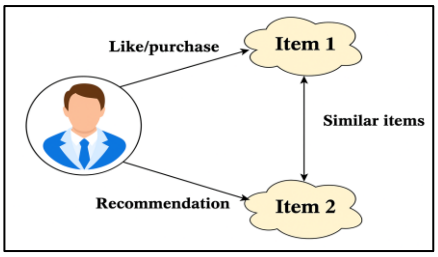


Figure 7: The content-based filtering.
Source [24].

3.2.1 Techniques Commonly Used in CBF

1. TF-IDF

In the domain of information retrieval, Term Frequency–Inverse Document Frequency (TF-IDF) is one of the most widely used techniques for weighting the importance of terms in a document. It serves as a fundamental weighting scheme that quantifies the importance of words based on their occurrence within documents. TF-IDF is frequently used in tasks such as keyword extraction, document similarity assessment, and relevance ranking [25].

In the approach presented, three statistical measures, Term Frequency (TF), Inverse Document Frequency (IDF), and their combination TF-IDF,are computed for each word token across both document clusters and individual texts.

- Term Frequency (TF) quantifies how often a specific term appear within a document, offering insight into the term's importance in that context. A higher TF score indicates greater significance of the term within the document. TF is formally defined as in equation (15)[25]:

$$TF_{t,d} = \frac{f_{t,d}}{max\{f_{t',d}:t'\in d\}} \cdot \qquad (15)$$

Where $f_{t,d}$ refers to the frequency of term t in document d, and the denominator
represents the maximum frequency of any term in the same document.

- Inverse Document Frequency (IDF), by contrast, measures how uncommon or distinctive a term is across a corpus. Words that appear in fewer documents receive higher IDF values, highlighting their discriminative power. It is defined as in formula (16) [25]:

$$IDF_{t,D} = log \frac{D}{|\{d\in D: t\in d\}|} \cdot \qquad (16)$$

where D is the total number of documents, and the denominator counts the number of documents in which the term t appears.

- The combined TF-IDF score is obtained by multiplying the TF and IDF values. This score increases when a term appears frequently in a particular document but rarely across the rest of the corpus, thereby indicating its relevance as in equation (17) [25]:

$$TF - IDF = TF_{t,d} \times IDF_{t,D} \cdot \qquad (17)$$

This weighting scheme effectively balances term frequency with term uniqueness,
making it a robust feature for tasks such as content-based recommendation and text classification.

2. Cosine Similarity

In content-based filtering, cosine similarity is commonly employed to measure the similarity between items based on their descriptive features. Each item is represented as a feature vector, such as keywords, genres, or TF-IDF scores, and the cosine of the angle between two such vectors is computed to assess how closely they are related. A cosine similarity score near 1 indicates a high degree of similarity between items, while a score closes to 0 reflects minimal or no similarity. This method is widely used to assess the similarity between any type of objects. Unlike other distance metrics, cosine similarity focuses on the orientation difference between two vectors, rather than a distance measure.

Cosine similarity plays a central role in identifying items that are similar to those items a user has previously interacted with, thereby enabling personalized recommendations based on item content. The similarity between two items a and b is calculated using the following formula (18)[12]:

$$Cosine\ similarity\ (a,b) = \frac{(a.b)}{||a||.||b||} \cdot \qquad (18)$$

3.2.2 The Process of Content-Based Filtering (CBF)

The content-based filtering technique operates through a structured and systematic process designed to generate personalized recommendations, as shown in Figure 8. This process consists of three essential stages [5]:

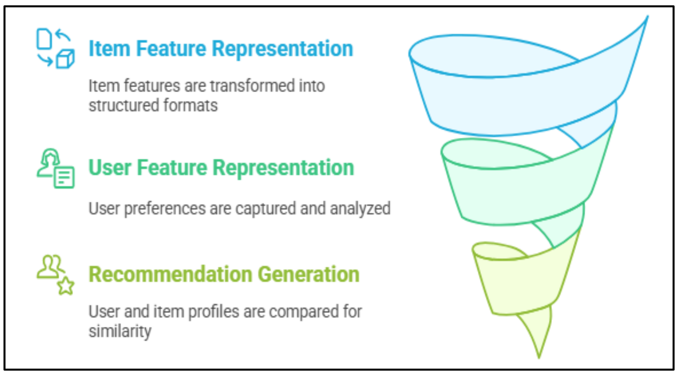

Item Feature Representation
Item features are transformed into structured formats
User Feature Representation
User preferences are captured and analyzed
Recommendation Generation
User and item profiles are compared for similarity

Figure 8: Content-based filtering stages.

1. Item Feature Representation

The first step involves constructing detailed representations of items based on their inherent attributes. These attributes may include descriptive metadata such as categories, tags, keywords, genres, and other relevant features that define the items within the recommendation domain. By transforming these features into structured formats (e.g., feature vectors), the system facilitates efficient comparison and similarity assessment between items.

2. User Feature Representation

Next, a user profile is developed to capture the user's preferences and behavioral patterns. This profile is typically derived from past user interactions, such as clicks, ratings, purchases, and search history. These interactions are analyzed to determine the user's favored item features, which are then incorporated into the profile to reflect the user's interests and preferences accurately.

3. Recommendation Generation

In the final step, the recommendation engine compares the user profile with the available item profiles using similarity computation methods, such as cosine similarity. Items that exhibit the highest degree of similarity to the user's preferences are ranked and presented as recommendations. This alignment ensures that the recommended items are highly relevant to the user's identified interests.

Overall, content-based filtering provides personalized recommendations by leveraging the relationship between item attributes and user preferences.

3.3 Hybrid Filtering

Initially, recommendation system was primarily utilized with single-method models, such as content-based filtering (CBF) and collaborative filtering (CF) being the most widely adopted approaches. However, these methods faced significant limitations. Content-based systems struggled with overspecialization as recommendations tend to focus narrowly on items similar to those already known to the user, while collaborative filtering methods suffered from the cold start problem (when there is insufficient data about new users or items) and data sparsity (when user interactions are limited).

Hybrid filtering emerged in the late 1990s to address these challenges, marking a pivotal shift in recommendation systems research. Balabanović and Shoham [26] were the first proponents of integrating collaborative and content-based filtering methods to mitigate individual weaknesses [26]. Over the years, hybrid recommendation systems attracted significant attention from researchers and emerged as one of the three most popular methods for generating suggestions in recommendation methods[27]. This popularity stems from their ability to combine two or more recommendation techniques, providing a more comprehensive approach to capturing user preferences.

Hybridization can also mitigate the weaknesses of individual methods, resulting in a more robust recommendation framework. Figure 9 illustrates the concept of a hybrid approach that combines collaborative filtering and content-based Filtering.

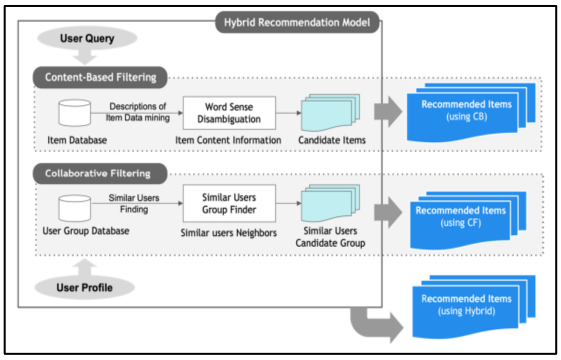

Hybrid Recommendation Model
User Query
Content-Based Filtering
Descriptions of Item Data mining
Word Sense Disambiguation
Item Database
Item Content Information
Candidate Items
Recommended Items (using CB)
Collaborative Filtering
Similar Users Finding
Similar Users Group Finder
User Group Database
Similar users Neighbors
Similar Users Candidate Group
Recommended Items (using CF)
User Profile
Recommended Items (using Hybrid)

Figure 9: hybrid filtering method
Source: [13].

Hybrid filtering can operate in several ways. For instance, it may integrate content-based filtering (CBF) into a collaborative filtering (CF) model, predict CBF and CF outcomes independently and then combine the results, or unify the systems into a single model [2].

3.3.1 Hybrid Filtering Types

Hybrid recommendation models integrate multiple filtering strategies to address the limitations of individual approaches and enhance overall performance. As outlined by Ko, et al. [13], hybrid models are divided into seven categories according to the specific strategies used to combine filtering methods. These categories are summarized as follows:

- **Weighted Hybridization** – Combines the outputs of different recommendation techniques by assigning weights, which are gradually adjusted according to the accuracy of predictions.
- **Switching Hybridization** – Dynamically switches between different recommendation strategies based on situational factors, such as data availability or user context.
- **Mixed Hybridization** – Generates recommendations by simultaneously combining results from multiple approaches, addressing challenges such as the cold-start problem in new items.
- **Feature Combination** – Merges item features and sample data, which are then processed by collaborative filtering, while content-based filtering operates on the enriched dataset.
- **Cascade Hybridization** – Applies one recommendation model to generate an initial candidate list of items, which is subsequently re-ranked by another model to refine accuracy.
- **Feature Augmentation** – Uses the output of one recommendation model (e.g., predicted scores or classifications) as input features for another recommendation model.
- **Meta-Level Hybridization – Employs the complete learned model from one approach as input to** another, allowing the meta-level representation of user preferences to enhance collaboration.

IV. Deep learning-based recommendation systems

Deep learning (DL) is a subfield of machine learning that uses artificial neural networks to enable machines to learn in a way that mimics human cognitive processes. It involves neural networks that are composed of more than three layers, allowing systems to learn complex patterns and representations [28]. It can automatically extract complex patterns from large-scale, high-dimensional, and unstructured data sources without relying on manual feature engineering [5, 29]

Deep learning has significantly advanced recommendation systems by enabling the modelling of complex, non-linear relationships between users and items. Unlike traditional linear models, deep learning architectures such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Autoencoders, and Transformer-based models allow for a deeper understanding of implicit and explicit user feedback. These models are particularly effective in environments where user-item interactions are sparse, high-dimensional, or highly contextual [30]. Deep learning models in a recommendation system can be classified into two categories as follow:

- **Non-Linear Learning in Recommendation Systems**

Deep learning models capture intricate patterns in user behavior and item features by employing non-linear activation functions (e.g., ReLU, Sigmoid, Tanh) across multiple layers. This enables the system to learn more abstract representations of preferences and item characteristics than linear models. For example, Neural Collaborative Filtering (NCF) replaces inner product operations with multi-layer perceptrons (MLPs), allowing for more expressive interactions (Alam & Ahmed, 2024). Furthermore, models such as Deep Neural Networks (DNNs) in Wide & Deep architectures facilitate the learning of both memorization and generalization patterns, essential for balancing long-term user behavior with recent interests [30].

- **Embeddings for Latent Feature Representation**

A cornerstone of deep learning-based recommendation systems is the use of embeddings. Embeddings are dense, low-dimensional vectors that encode semantic similarities between users and items. These vectors are learned during training and help uncover latent factors influencing user preferences.

For instance, DeepFM integrates Factorization Machines with deep neural networks to capture both low- and high-order feature interactions in recommendation tasks [30].. Similarly, models such as AutoRec use autoencoders to reconstruct user-item matrices, improving performance in cold-start and sparsity scenarios by capturing non-linear dependencies [30].

V. Evaluation Methods of Recommendation Systems

Evaluating recommender systems is a crucial step in determining their effectiveness, robustness, and practical applicability. This section outlines the key methods used to assess recommender systems, which are organized into three main components:

5.1 Evaluation Protocols

For evaluation, datasets are typically split into three parts: training, validation, and test sets. The training set is used to fit the model, while the validation set helps fine-tune the model by adjusting hyperparameters and preventing overfitting through techniques like early stopping. The model's final performance is then assessed using the test set. In some situations, only training and test sets are used without a separate validation set [31].

5.2 Evaluation Metrics

Since the inception of research on recommender systems (RS), evaluating the accuracy of predictions and recommendations has become increasingly important. To assess the effectiveness of various recommendation techniques, methods, and algorithms, the field relies on specific quality indicators and performance metrics. The choice of evaluation metrics depends not only on the filtering approach used but also on the characteristics of the dataset and the nature of the recommendation tasks being performed. The most widely adopted metrics include:

1. Mean Absolute Error (MAE)

MAE is a commonly employed evaluation metric in recommender systems. It measures the average absolute difference between predicted ratings and the actual ratings assigned by users. A smaller MAE value signifies higher accuracy in capturing user preferences. The computation of MAE is expressed by formula (19) [24]:

$$\text{MAE} = \frac{1}{N} \sum_{u,i} | \, p_{u,i} - r_{u,i} \, | \,. \qquad (19)$$

Here, $p_{u,i}$ is the predicted rating for user *u* and item *i*, $r_{u,i}$ is the actual rating, and *N* is the total number of predictions.

2. Root Mean Square Error (RMSE)

RMSE is a statistical accuracy metric commonly used in evaluating recommender systems. Unlike MAE, RMSE focuses more on larger absolute errors, making it particularly sensitive to outliers or poor predictions. As a result, it tends to yield higher values than MAE. A lower RMSE indicates that the recommendation model provides more precise predictions. The RMSE is calculated using the formula (20) [24]:

$$\text{RMSE}= \sqrt{\frac{1}{N} \sum_{u,i} \left(r_{u,i} - \hat{r}_{u,i}\right)^2} \,. \qquad (20)$$

Where $r_{u,i}$ is the actual rating, $\hat{r}_{u,i}$ is the predicted rating, and *N* is the total number of user-item pairs evaluated.

3. Precision and Recall

Precision refers to the proportion of recommended items that are relevant to the user, while recall measures the proportion of the relevant items that have been successfully recommended out of all relevant items that should have been recommended [32]. In this context, a relevant item aligns with the user's preferences. When evaluating a classification model, Precision and Recall are two key metrics used to understand how well your model is performing, particularly when dealing with imbalanced data or binary classification problems [33]. These metrics are derived from the confusion matrix, which summarizes the results of predictions versus actual outcomes. Table 3 presents a confusion matrix to illustrate how Precision and Recall are calculated. The confusion matrix outlines four possible outcomes of a recommendation. If a recommended item is relevant to the user, it is counted as a True Positive (TP); if a relevant item is not recommended, it is a False Negative (FN). Conversely, recommending an irrelevant item results in a False Positive (FP), while correctly not recommending an irrelevant item is a True Negative (TN).

Table 2
A confusion matrix for a recommendation system.

	Predicted Positive	**Predicted Negative**
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

In this confusion matrix:
Precision measures the proportion of correctly predicted positive instances among all predicted positives [34]. High precision score indicates fewer false positives, making it useful in systems where incorrect recommendations can negatively affect user experience or trust [35, 36]. It is calculated as in equation (21):

$$\text{Precision} = \frac{TP}{TP+FP} \ . \qquad (21)$$

Where TP is True Positive (TP) and FP False Positive (FP). Recall measures the ability of the system to identify all relevant items [34]. A high recall score indicates that the system successfully retrieves the most relevant items [37, 38]. It is computed as in formula (22):

$$\text{Recall} = \frac{TP}{TP+FN} \ . \qquad (22)$$

Where TP is True Positive (TP) and FN is False Negative (FN).

4. F-measure (F1-score)

The F1-score is an evaluation metric that integrates Precision and Recall into a single measure. Precision denotes the proportion of recommended items that are relevant, whereas Recall represents the proportion of relevant items successfully recommended. The F1-score, also known as the F-measure, is calculated as the harmonic mean of these two metrics, ensuring that the value is high only when both Precision and Recall are sufficiently strong [24, 39]. This metric is particularly valuable in scenarios where minimizing both false positives and false negatives is crucial, such as spam detection or personalized content recommendation [39, 40]. Its computation is given in formula (23):

$$\text{F1} - \text{score} = \frac{Precision \times Recall}{Precision + Recall} \ . \qquad (23)$$

5. Accuracy Matrix

Accuracy is among the simplest and most widely applied metrics for assessing the effectiveness of classification and recommendation systems. It represents the ratio of correctly predicted instances—encompassing both true positives and true negatives—to the total number of predictions, as defined in Equation (24) or (25):

$$\text{Accuracy} = \frac{Number\ of\ correct\ recomendations}{Total\ number\ o\ f\ recommendations} \ . \qquad (24)$$

Or:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \ . \qquad (25)$$

Where, TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives.

6. Normalized Discounted Cumulative Gain (NDCG)

NDCG is a widely used evaluation metric in recommendation systems and information retrieval, especially in ranking-based tasks. It incorporates both graded relevance (i.e., the value of user ratings) and the position of recommended items in a ranked list. It measures how well a recommendation system ranks items based on both the relevance of recommended items and their position in the recommendation list. [41]. The NDCG is calculated as follows equation (26) [42]:

$$\text{NDCG@k} = \frac{1}{IDCG@k} \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{log_2(i+1)} \ . \qquad (26)$$

In this formula, rel_i denotes the graded relevance score (e.g., a user rating) of the item at position i in the recommended list. The numerator represents the Discounted Cumulative Gain (DCG), which accounts for the position of relevant items using a logarithmic discount to prioritize higher-ranked results. The denominator, IDCG@k, is the Ideal DCG and reflects the maximum possible DCG obtainable from a perfectly ranked list. By normalizing DCG with IDCG, the resulting NDCG@k score ranges between 0 and 1, where a value of 1 indicates a perfectly ranked recommendation list.

5.3 Comparison with Existing Studies

To situate this research within the broader scholarly landscape, it is essential to compare the selected evaluation methods and findings with those of existing studies. Such comparisons highlight methodological similarities and differences, reveal gaps in current research, and demonstrate how the present work advances knowledge in the field of recommender systems. To ensure fairness and reliability, this study compares its results with prior research that employed the same datasets and evaluation metrics, thereby enhancing the validity of cross-study comparisons and ensuring methodological consistency.

VI. The Common Challenges in RSs

Recommendation systems, while pivotal in enhancing user experience and engagement across various platforms, face several persistent challenges that impact their effectiveness and scalability. The primary challenges faced by recommendation systems are concisely presented below

- **Cold Start Issues**

The cold start problem arises when a recommendation system must handle new users or items without any prior interaction data, which considerably undermines the effectiveness of collaborative filtering (CF) methods [2]. This challenge typically appears in three cases: (a) the arrival of a new user, (b) the addition of a new item, or (c) the creation of a new user group or community. Because CF relies extensively on historical user–item interactions, it faces difficulties in generating reliable recommendations when such information is unavailable.

The cold start problem arises when a recommendation system must handle new users or items without any prior interaction data, which considerably undermines the effectiveness of collaborative filtering (CF) methods [2, 24]. This challenge typically appears in three cases: (1) the arrival of a new user, (2) the addition of a new item, or (3) the creation of a new user group or community. Because CF relies extensively on historical user–item interactions, it faces difficulties in generating reliable recommendations when such information is unavailable[24].

- **Data Sparsity Issue**

Data sparsity is major challenge in developing effective recommendation systems. This issue arises when the user–item interaction matrix, employed in collaborative filtering (CF), contains far fewer observed interactions than the total number of potential ones [43]. In large-scale systems, users typically interact with only a limited subset of items, producing a highly sparse matrix with insufficient information to capture meaningful relationships between users and items [44]. Such sparsity severely hampers the performance of CF algorithms, which rely on past interactions to compute user–user or item–item similarities. When most matrix entries remain unfilled, estimating these similarities becomes unreliable, often resulting in weak or irrelevant recommendations [45].

- **Scalability**

Scalability in recommender systems denotes the capacity to preserve efficiency, responsiveness, and accuracy as the numbers of users, items, and interactions increase. In large-scale digital settings, such as e-commerce and streaming platforms, scalability ensures that recommendations remain timely and relevant without compromising system performance. However, traditional approaches, particularly those grounded in collaborative filtering and matrix factorization, often struggle to scale effectively due to heightened computational demands and the challenges posed by data sparsity [4].

- **Over-Specialization Issue**

Over-specialization in recommender systems occurs when the system repeatedly recommends items that are too similar to those a user has already preferred, thereby reducing diversity and novelty in the suggestions. This problem is especially pronounced in Content-Based Filtering (CBF), since such methods depend primarily on item feature similarities.[46].

- **Grey Sheep Issue**

The grey sheep problem arises in recommender systems when certain users exhibit unpredictable or inconsistent preferences that differ significantly from the majority. Unlike typical users who consistently align with identifiable behavior patterns, grey sheep users occasionally agree with many groups but do not closely match any particular one [47, 48]. This inconsistency makes it difficult for collaborative filtering (CF) algorithms, especially those based on user similarity, to produce accurate recommendations for these users.

- **Diversity Issue**

In many cases, recommender systems may provide suggestions that are either closely related or deliberately diverse. However, the most accurate predictions often emerge from emphasizing similarities among users or items, which gives rise to the **diversity problem**—recommendations concentrate on commonalities while overlooking differences. Consequently, users are exposed to a narrow range of content and may miss less popular but highly relevant niche items.

This issue is particularly evident in collaborative filtering (CF) systems. Since CF depends on historical user–item interactions, it tends to reinforce the popularity of frequently rated or consumed items, resulting in many users receiving similar popular recommendations. This dynamic produces popularity bias, whereby niche or long-tail items are marginalized despite their potential relevance to individual users [49, 50]. As a result, users often encounter a restricted set of items, limiting opportunities for discovery and personalization. Such effects can undermine both user satisfaction and fairness, particularly for individuals with distinctive preferences or those seeking non-mainstream content.

VII. Conclusion and future work

Recommender systems (RSs) have become indispensable in addressing the challenge of information overload by providing users with personalized and relevant suggestions. This paper reviewed the historical development of RSs, traditional methods such as collaborative and content-based filtering, as well as the emergence of hybrid approaches. It also highlighted the transformative role of deep learning models, which have significantly advanced the capacity of RSs to capture complex user–item relationships and integrate multimodal data. Furthermore, evaluation methods and metrics were discussed, underscoring their importance in systematically assessing the effectiveness of recommendation models. The study also synthesized the key challenges currently facing RSs, including data sparsity, scalability, diversity, over-specialization and cold-start issues. In future work, researchers plan to review cross-domain and transfer learning approaches in recommender systems, focusing on how knowledge transfer across domains can improve personalization and mitigate challenges such as data sparsity and the cold-start problem.

VIII.Acknowledgement

The authors would like to express sincere appreciation to the Faculty of Computer & Information Technology, Al-Madinah International University, Kuala Lumpur, Malaysia, for their support in this research. Gratitude is also extended to the anonymous reviewers for their valuable comments and constructive feedback.

IX. Declaration of Computing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

X. REFERENCES

[1] F. Ricci, L. Rokach, and B. Shapira, "Recommender systems: Techniques, applications, and challenges," *Recommender systems handbook,* pp. 1-35, 2021.

[2] D. Roy and M. Dutta, "A systematic review and research perspective on recommender systems," *Journal of Big Data,* vol. 9, no. 1, p. 59, 2022.

[3] M. A. Hodovychenko and A. A. Gorbatenko, "Recommender systems: models, challenges and opportunities," *HAIT,* vol. 6, no. 4, pp. 308-319, 2023.

[4] X. Zhang, "Research on Intelligent Recommendation Algorithm Based on Deep Learning," *International Journal of Computer Science and Information Technology,* 2024.

[5] H. Zhou, F. Xiong, and H. Chen, "A comprehensive survey of recommender systems based on deep learning," *Applied Sciences,* vol. 13, no. 20, p. 11378, 2023.

[6] H. D. Tran, "Towards a new generation of deep learning based recommender systems," Macquarie University, 2021.

[7] N. J. Belkin and W. B. Croft, "Information filtering and information retrieval: Two sides of the same coin?," *Communications of the ACM,* vol. 35, no. 12, pp. 29-38, 1992.

[8] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Communications of the ACM,* vol. 35, no. 12, pp. 61-70, 1992.

[9] Z. Dong, Z. Wang, J. Xu, R. Tang, and J. Wen, "A brief history of recommender systems," *arXiv preprint arXiv:2209.01860,* 2022.

[10] K. Bhareti, S. Perera, S. Jamal, M. H. Pallege, V. Akash, and S. Wiieweera, "A literature review of recommendation systems," in *2020 IEEE International Conference for Innovation in Technology (INOCON)*, 2020: IEEE, pp. 1-7.

[11] I. Saifudin and T. Widiyaningtyas, "Systematic Literature Review on Recommender System: Approach, Problem, Evaluation Techniques, Datasets," *IEEE Access,* 2024.

[12] S. C. Mana and T. Sasipraba, "Research on cosine similarity and pearson correlation based recommendation models," in *Journal of Physics: Conference Series*, 2021, vol. 1770, no. 1: IOP Publishing, p. 012014.

[13] H. Ko, S. Lee, Y. Park, and A. Choi, "A survey of recommendation systems: recommendation models, techniques, and application fields," *Electronics,* vol. 11, no. 1, p. 141, 2022.

[14] W. Zhao, H. Tian, Y. Wu, Z. Cui, and T. Feng, "A new item-based collaborative filtering algorithm to improve the accuracy of prediction in sparse data," *International Journal of Computational Intelligence Systems,* vol. 15, no. 1, p. 15, 2022.

[15] U. Sultan, H. Khan, Y. S. Afridi, M. I. A. Shah, M. Ashfaq, and A. Sultan, "Performance Analysis of a Hybrid Recommender System," 2024.

[16] F. Fkih, "Similarity measures for Collaborative Filtering-based Recommender Systems: Review and experimental comparison," *Journal of King Saud University-Computer and Information Sciences,* vol. 34, no. 9, pp. 7645-7669, 2022.

[17] J. A. Morozov and S. E. Saradgishvili, "Improving Collaborative Filtering," *2021 IV International Conference on Control in Technical Systems (CTS),* pp. 232-234, 2021.

[18] A. Iftikhar, M. A. Ghazanfar, M. Ayub, Z. Mehmood, and M. Maqsood, "An improved product recommendation method for collaborative filtering," *IEEE Access,* vol. 8, pp. 123841-123857, 2020.

[19] W. X. Zhao, Z. Lin, Z. Feng, P. Wang, and J.-R. Wen, "A revisiting study of appropriate offline evaluation for top-N recommendation algorithms," *ACM Transactions on Information Systems,* vol. 41, no. 2, pp. 1-41, 2022.

[20] W. Nguyen, "A Literature Review of Collaborative Filtering Recommendation System using Matrix Factorization algorithms," in *Proceedings of ACM Conference (Conference'17). ACM*, 2021.

[21] S. Loeb and D. Terry, "Information filtering," *Commun. ACM,* vol. 35, no. 12, pp. 26–28, 1992, doi: 10.1145/138859.138860.

[22] P. Duraisamy, S. Yuvaraj, Y. Natarajan, and V. Niranjani, "An overview of different types of

recommendations systems-a survey," in *2023 4th International Conference on Innovative Trends in Information Technology (ICITIIT)*, 2023: IEEE, pp. 1-5.

[23] U. Javed, K. Shaukat, I. A. Hameed, F. Iqbal, T. M. Alam, and S. Luo, "A review of content-based and context-based recommendation systems," *International Journal of Emerging Technologies in Learning (iJET),* vol. 16, no. 3, pp. 274-306, 2021.

[24] D. Patel, F. Patel, and U. Chauhan, "Recommendation Systems: Types, Applications, and Challenges," *International Journal of Computing and Digital Systems,* 2023.

[25] W. I. Al-Obaydy, H. A. Hashim, Y. Najm, and A. A. Jalal, "Document classification using term frequency-inverse document frequency and K-means clustering," *Indonesian Journal of Electrical Engineering and Computer Science,* vol. 27, no. 3, pp. 1517-1524, 2022.

[26] M. Balabanović and Y. Shoham, "Fab: content-based, collaborative recommendation," *Commun. ACM,* vol. 40, no. 3, pp. 66–72, 1997, doi: 10.1145/245108.245124.

[27] B. Sabiri, A. Khtira, B. El Asri, and M. Rhanoui, "Hybrid Quality-Based Recommender Systems: A Systematic Literature Review," *Journal of Imaging,* vol. 11, no. 1, p. 12, 2025.

[28] L. Salau, M. Hamada, R. Prasad, M. Hassan, A. Mahendran, and Y. Watanobe, "State-of-the-art survey on deep learning-based recommender systems for e-learning," *Applied Sciences,* vol. 12, no. 23, p. 11996, 2022.

[29] S. K. Mohanta, A. G. Mohapatra, A. Mohanty, and S. Nayak, "Deep Learning is a State-of-the-Art Approach to Artificial Intelligence," in *Deep Learning Concepts in Operations Research*: Auerbach Publications, 2024, pp. 27-43.

[30] M. M. Alam and M. Ahmed, "Deep Learning-Based Recommendation Systems: Review and Critical Analysis," in *International Conference on Data Analytics & Management*, 2023: Springer, pp. 39-55.

[31] X. Xin, "Deep learning-based implicit feedback recommendation," University of Glasgow, 2021.

[32] Z. Fayyaz, M. Ebrahimian, D. Nawara, A. Ibrahim, and R. Kashef, "Recommendation systems: Algorithms, challenges, metrics, and business opportunities," *applied sciences,* vol. 10, no. 21, p. 7748, 2020.

[33] J. Miao and W. Zhu, "Precision–recall curve (PRC) classification trees," *Evolutionary intelligence,* vol. 15, no. 3, pp. 1545-1569, 2022.

[34] P. Fränti and R. Mariescu-Istodor, "Soft precision and recall," *Pattern Recognition Letters,* vol. 167, pp. 115-121, 2023.

[35] J. Govea, R. Gutierrez, and W. Villegas-Ch, "Transparency and precision in the age of AI: evaluation of explainability-enhanced recommendation systems," *Frontiers in Artificial Intelligence,* vol. 7, p. 1410790, 2024.

[36] J. Cook and V. Ramadas, "When to consult precision-recall curves," *The Stata Journal,* vol. 20, no. 1, pp. 131-148, 2020.

[37] W. Kusa, G. Peikos, M. Staudinger, A. Lipani, and A. Hanbury, "Normalised precision at fixed recall for evaluating tar," in *Proceedings of the 2024 ACM SIGIR International Conference on Theory of Information Retrieval*, 2024, pp. 43-49.

[38] F. Diaz, M. D. Ekstrand, and B. Mitra, "Recall, robustness, and lexicographic evaluation," *ACM Transactions on Recommender Systems,* 2025.

[39] Z. Ding, "Design and evaluation of an English speaking recommender system using word networks and context-aware techniques," *Entertainment Computing,* vol. 52, p. 100920, 2025.

[40] J. C. Obi, "A comparative study of several classification metrics and their performances on data," *World Journal of Advanced Engineering Technology and Sciences,* vol. 8, no. 1, pp. 308-314, 2023.

[41] D. Valcarce, A. Bellogín, J. Parapar, and P. Castells, "Assessing ranking metrics in top-N recommendation," *Information Retrieval Journal,* vol. 23, pp. 411-448, 2020.

[42] C. Li, I. Ishak, H. Ibrahim, M. Zolkepli, F. Sidi, and C. Li, "Deep learning-based recommendation system: systematic review and classification," *IEEE Access,* vol. 11, pp. 113790-113835, 2023.

[43] L. Li, Z. Zhang, and S. Zhang, "Hybrid algorithm based on content and collaborative filtering in recommendation system optimization and simulation," *Scientific Programming,* vol. 2021, no. 1, p. 7427409, 2021.

[44] Y. Afoudi, M. Lazaar, and M. Al Achhab, "Hybrid recommendation system combined content-based filtering and collaborative prediction using artificial neural network," *Simulation Modelling Practice and Theory,* vol. 113, p. 102375, 2021.

[45] G. Parthasarathy and S. Sathiya Devi, "Hybrid recommendation system based on collaborative and content-based filtering," *Cybernetics and Systems,* vol. 54, no. 4, pp. 432-453, 2023.

[46] O. Stitini, S. Kaloun, and O. Bencharef, "An improved recommender system solution to mitigate the over-specialization problem using genetic algorithms," *Electronics,* vol. 11, no. 2, p. 242, 2022.

[47] G. Bejaoui and R. Ayachi, "A trust-based clustering approach for identifying grey sheep users," in *Digital Economy. Emerging Technologies and Business Innovation: 5th International Conference on Digital Economy, ICDEc 2020, Bucharest, Romania, June 11–13, 2020, Proceedings 5*, 2020: Springer, pp. 76-88.

[48] Al Moaiad, Y., Alobed, M., Alsakhnini, M., & Momani, A. M. (2024). Challenges in natural Arabic language processing. *Edelweiss Applied Science and Technology*, 8(6), 4700-4705.

[49] Al Moaiad, Y., D., Alkhateeb, M., & Alokla, M. (2024). Deep Analysis of Iris Print and Fingerprint for Detecting Drug Addicts. *J. Electrical Systems*, 20(3), 5639-5644.

[50] Al Moaiad, Y., D., Alkhateeb, M., & Alokla, M. (2024). Enhancing Cybersecurity Practices in Nigerian Government Institutions an Analysis and Framework. *J. Electrical Systems,* 20(3), 5964-5967.