Enhancing Cold-Start Recommendations with Content-Based Profiles and Latent Factor Models

Original Scientific Paper

Amritha P*
Kannur University,
Department of Information Technology,
Kannur University, Kannur, Kerala, India
amritha@chintech.ac.in

Rajkumar K K
Kannur University,
Department of Information Technology,
Kannur University, Kannur, Kerala, India
rajkumarkk@kannuruniv.ac.in

*Corresponding author

Abstract *– Recommendation systems have become an important tool for enhancing personalized recommendations across various domains. However, these systems face challenges, including the cold start problem, data sparsity, etc. In this paper, we present a novel recommendation model that integrates content-based and collaborative approaches to overcome these challenges. The proposed model uses TF-IDF vectorization over multiple item attributes to compute content similarity scores, and the SVD collaborative model captures latent user-item interactions. To further strengthen user preferences, a time-aware exponential decay function is used to acquire the most recent user preferences during the construction of user profiles for content-based prediction. Finally, the rating prediction is generated through a weighted fusion of content and collaborative models. Compared to benchmark models, our approach reduces RMSE by 3.06% and MAE by 3.23%, demonstrating an improvement in prediction accuracy. Furthermore, our method shows stable performance, with only a slight increase in prediction error (MAE with 8% and RMSE with 1.5% with a hybrid weight of 0.5) under cold-start conditions, indicating that the proposed method maintains strong stability and robustness even in data sparsity scenarios.*

Keywords: *recommendation, collaborative, hybrid, content-based*

Received: June 18, 2025; Received in revised form: October 1, 2025; Accepted: October 2, 2025

1. INTRODUCTION

The amount of data over the Internet has increased dramatically in the past few years due to the rapid advancement of information technology. Although the Internet offers more accessibility to users, it also creates the issue of "information overload" [1]. This is a big challenge for consumers to easily and precisely identify the necessary information among the enormous volume of data. Recommendation systems (RS), or recommender systems, are essential tools for consumers to find required personalized details from the internet. In recent years, academics and industry have focused on recommendation systems, which effectively help alleviate the problem of information overload [1, 2].

User-specific collections generated by recommendation systems make exploring the internet a satisfying experience for customers. Recommendation systems consist of three primary categories: content-based methods, collaborative filtering approaches, and hybrid models that incorporate both approaches [3]. Content-based filtering creates recommendations by analyzing the features or metadata of items, such as genre, keywords, or descriptions, and aligning them with the known preferences of users [3]. By identifying patterns and relationships among users and items, collaborative filtering can recommend items. Collaborative filtering is the most commonly employed algorithm in recommendation systems. Based on past behaviour, it predicts user preferences and generates customized recommendations [1]. Hybrid recommendation systems merge collaborative and content-based systems to provide better suggestions [4]. Recommendation systems typically produce two kinds of outputs: (a) a list of the top N recommended things, and (b) a numerical prediction for a user or set of users.

Recommendation systems help to reduce information overload by giving personalized suggestions, but they still face many challenges [1, 5]. One of the major issues among them is the data sparsity problem, which is a situation where the available user-item interaction data is sparse. This can occur when there are many users and items in the system, but each user has not interacted or given feedback. Another challenge to be addressed is the cold-start problem, which arises when newly introduced items are not getting listed in the recommended item list. Therefore, researchers have been trying to improve these algorithms and explore other techniques and methods to solve these problems and improve the effectiveness, accuracy, and user satisfaction of recommendation systems. The dynamic interests of customers also affect the efficiency of recommendations. For example, people may have long-term interests and short-term interests based on different contexts. Traditional recommendation methods cannot address these kinds of problems. Addressing these challenges requires innovative approaches, including hybrid models that combine different recommendation techniques, the incorporation of contextual and real-time data, and the development of mechanisms to enhance privacy, fairness, and transparency in recommendation processes [2, 3]. Content-based techniques focus on analyzing the item attributes that a user has interacted with and provide suggestions based on similarities between the user's preferences and item features. However, content-based methods do not incorporate user behaviour data, which means they cannot adapt to users' changing preferences. At the same time, collaborative filtering methods use user behaviour data, such as ratings, clicks, and other types of interactions [5].

So incorporating auxiliary data, contextual data, temporal features and user preferences is essential for enhancing the content-based methods, while collaborative approaches are vulnerable to data sparsity since many users do not consistently provide ratings. Traditional methods have their advantages and limitations, which vary depending on the application context. To exploit the strengths of both content-based and collaborative filtering while addressing these shortcomings, hybrid strategies with additional attributes are essential for tailoring recommendations to customers [4, 5].

The main contributions of this study are summarized as follows:

1. This study introduces a hybrid recommendation model that integrates content-based and collaborative filtering techniques to effectively address cold-start challenges, including user cold-start and item cold-start.
2. Our method uses the vectorization of multiple item features in the content-based component to make meaningful predictions, even in the absence of historical user-item interactions. The collaborative part uses Singular Value Decomposition (SVD) to uncover hidden patterns in the user-item rating matrix.
3. These methods incorporate an additional time-aware exponential decay function to capture the item's timestamp feature, which is used for preparing the user profile. This allowed the system to accord greater emphasis to more recently rated items, enhancing the relevance of the user profile preferences.

The following sections of this article are organized as follows: Section 2 provides an introduction about recommendation systems and their types. In Section 3, we discuss the related works on recommendation systems. In Section 4, we outline our proposed recommendation model and methodology. Finally, in the last part, we present the findings of our experimental results and the implications of our study.

2. RECOMMENDATION ALGORITHMS

The two main sections of recommendation systems are content filtering and collaborative filtering. Each uses a different set of techniques to offer items to the user. Collaborative filtering uses user-item interactions or ratings to discover correlations between customers and items. Content-based filtering, in contrast, looks at the characteristics of the items and recommends items similar to what the user has liked before, relying mainly on the description and features of the items [3].

2.1. COLLABORATIVE-BASED RECOMMENDATION

The fundamental input for the collaborative techniques is a user-item rating matrix. The recommendation system assumes that users will have the same preferences in the future if they liked or interacted with similar items. Two major categories of collaborative filtering are memory-based and model-based.

The three basic phases of memory-based collaborative recommendation systems are: 1) calculating similarity; 2) identifying nearest neighbours among similar users/items; and 3) making predictions. Two important subfields of memory-based collaborative filtering recommendation systems are the item-based and user-based approaches. Model-based techniques use mathematical models to learn hidden features of the user-item rating matrix that represent users and items in a lower-dimensional space. These models extract latent features from the user interaction matrix to identify underlying patterns in the data [2, 3]. One of the popular models based on the collaborative filtering method is singular value decomposition [6].

The SVD is a widely recognized matrix factorization (MF) technique in recommendation systems. Its main purpose is to reduce the dimensionality of the user-item rating matrix while preserving important relationships. The notion of SVD involves reducing the dimensionality of the original matrix through its factorization into smaller, low-rank matrices, thereby capturing latent relationships. Consider a matrix A of size m x n is transformed into: U with size m x f, $\sum$ with size f x f, and V with size f x n, as shown in Fig. 1 [7, 8].

In recommendation systems, SVD approximates the rating matrix by decomposing it into two lower-rank matrices, P and Q. The matrix P corresponds to user-feature interactions derived by U x $\sum$, where $\sum$ is treated as a scalar value to preserve the dimensionality. The matrix Q represents item-feature interactions and is equivalent to V. The dot product of P and Q estimates the rating a user might assign to an unseen item [9]. Therefore, the SVD-based matrix factorization formulation can be expressed using the user-item rating matrix **R** as follows:

$$\mathbf{R}_{m\times n} \approx \mathbf{P}_{m\times f}(\mathbf{Q}_{n\times f})^T \qquad (1)$$

Where $R_{m \times m}$ is the user-item rating matrix. $P_{m \times f}$ denote user latent matrix (users × latent factors). And $Q_{f \times n}$ denote the Item latent matrix (items × latent factors). Fig. 2. depicted SVD-based decomposition of a user-item interaction/rating matrix. In this rating matrix, the blank cells represent missing ratings or values. A higher number of such blank cells indicates greater data sparsity [9]. Matrices P and Q are derived by factorizing the user-item rating matrix R. The matrix P denote a user-latent matrix, and matrix Q depicts an item-latent matrix. These two matrices are of m x f and f x n respectively. Here m denotes the number of users, n refers to the number of items, and f represents the number of latent factors obtained during matrix decomposition.

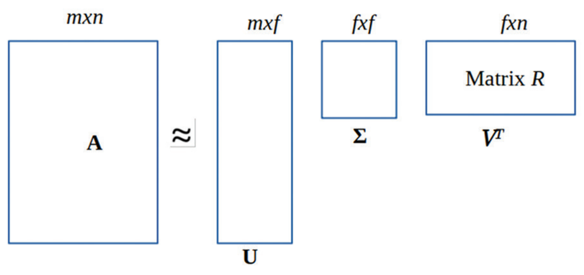

Fig. 1. SVD decomposition

The number of latent factors can be selected based on the required model complexity, as they help to uncover hidden patterns and interactions between users and items [8, 9]. The main advantage of the SVD method in recommendation systems is that it overcomes data sparsity and scalability problems. However, applying SVD directly to collaborative filtering can be problematic due to the presence of many missing entries in the user-item rating matrix. To perform matrix factorization, these missing values are often filled with some default values. Regularized SVD that works through iteration is called matrix factorization [9].

2.2. CONTENT-BASED (CB) RECOMMENDATION

The content-based approach includes a metadata of item characteristics and a user profile which contains the user's historical interests. The central task of this recommendation system is identifying items that closely match with the user's individual preferences [5]. Unlike collaborative methods, content-based recommendation depends on the inherent qualities of items and the user's preferences [5]. The Term Frequency-Inverse Document Frequency (TF-IDF), binary encoding, categorical encoding method, or the frequency encoding method are some of the techniques used to process the textual data in item descriptions in content-based recommendation systems [10, 11]. This model considers only the information provided by the target user and the features of the rated items for predicting the recommendation. Content-based algorithms use user preferences for items and suggest similar ones based on a domain-specific understanding of the item's content [12]. The initial step for a content-based model is textual data from item descriptions processed with a Vector Space Model (VSM) [12]. Below is a list of all the steps involved in the CB recommendation model process:

1. Initially each item's textual feature string is vectorized using TF-IDF to produce a sparse, high-dimensional feature representation.
2. In the second step, these vectors are used to compute *cosine similarity*, which measures the closeness between items. To personalize recommendations, construct a *user profile vector* by taking a weighted average of the TF-IDF vectors of the item the user has rated, where the weights are the actual rating values [8].
3. There are two ways to compute the predicted rating. a) Item-based similarity approach: For a given user and a target object/item, the system takes into account the items that the user has previously rated. It calculates the weighted average of those ratings, where each weight is the cosine similarity across the target item and a previously rated item. b) User profile method: The predicted rating for a new item is computed as the cosine similarity between the user's profile vector, which is constructed from the TF-IDF vectors of items they have rated, and the TF-IDF vector of the target item [8].

Let us take an example. Each movie's content description is formed by concatenating its title, genres, and release year (binned by decade). That is, the movie Toy Story (1995), with the genres Animation and Comedy, becomes the string "toy story animation comedy 90s". This is depicted in Table 1. These textual feature strings are transformed into numerical vectors using the TF-IDF method, which captures the importance of each term within the corpus. Once vectorized, the cosine similarity is computed between every pair of movies, resulting in a similarity matrix as in Table 2. This matrix reflects how semantically close each pair of movies is based on their textual features. These cosine similarity values are later used for generating content-based recommendations either by comparing item-item relationships or by a user profile vector method.

Table 1. Content Representation Used for TF-IDF Encoding

Movie	Title	Genres	Year	TF-IDF Content String
A	"Toy Story"	"Animation, Comedy"	1995	"toy story animation comedy 90s"
B	"The Lion King"	"Animation, Adventure"	1994	"lion king animation adventure 90s"
C	"Aladdin"	"Animation, Fantasy"	1992	Aladdin animation fantasy 90s"

Table 2. Cosine Similarity Matrix Based on TF-IDF Vectors

	Toy Story	The Lion King	Aladdin
Toy Story	1.000	0.189	0.221
The Lion King	0.189	1.000	0.221
Aladdin	0.221	0.221	1.000

Hybrid recommendation merges various strategies such as content-based and collaborative filtering. This approach allows for more personalized and accurate recommendations for users with diverse preferences and behaviours. The main advantage of hybrid recommendation systems is their ability to overcome the weaknesses of individual recommendation methods [3].

3. RELATED WORK

The development of hybrid recommendation systems has improved in recent years as researchers attempt to overcome the challenges of sparsity, cold start, popularity bias, and evolving user preferences. Traditional collaborative filtering methods are good at capturing latent user–item interactions but get worse in sparse conditions, while content-based filtering can handle new items but struggles with limited attributes and user personalization. To address these shortcomings, numerous hybridization strategies have been proposed, each introducing novel mechanisms but also new trade-offs in complexity, scalability, and effectiveness.

One way to strengthen CF under sparsity is by integrating memory based Nearest-Neighbour model with model based collaborative filtering. Lv et al. [13] proposed a hybrid recommendation algorithm that integrates a User-Nearest-Neighbour (UNN) model with collaborative filtering techniques. The novelty of this approach lies in using the UNN model to fill missing user–item interactions with a weighted similarity metric. After this step, the collaborative filtering methods called ALS, MLP, and NCF are applied on the optimized matrix. The key advantage of this method is that it can reduce sparsity and enhance accuracy in sparse situations. It uses the Spark distributed platform to make it scalable. However, the model is less effective when users have few co-interactions and does not address the item cold-start since it ignores content features [13]. Similarly, Guan et al. [14] came up with an advanced similarity computation with a Wasserstein-distance-based CF, integrating anti-popularity and anti-prominence terms to reduce bias. The main advantage of this work is in its ability to handle sparse datasets and its evaluation across different metrics. However, the method incurs higher computational cost due to the Wasserstein distance calculation and similarity-based CF without explicit incorporation of temporal or content information [14].

Another important direction is adaptive segmentation and neighborhood personalization. Liang et al. [15] proposed a behavior-aware hybrid recommendation framework that separates users into two groups: active groups and inactive groups. For the inactive users, the method designed a fusion algorithm that integrates SVD with content-based filtering, which improved the accuracy measures on the MovieLens dataset. For the active users, the method applied a diversity-enhanced KNN algorithm, which reduced accuracy but increased item coverage, thereby enhancing diversity. The positive aspects of this work is its explicit user-group differentiation, ensuring a solution for sparsity. While it balances accuracy and diversity, it does not explicitly address the cold-start problem, since new users and items are not the primary focus of the framework [15].

Roy et al. [7] took an alternate approach, proposing a weighted hybrid model that combined Adaptive KNN (AKNN) and SVD. AKNN used a hybrid similarity measure that integrates cosine similarity, Pearson correlation, and Variance Mean Difference (VMD). The SVD model is used to capture latent user and item factors through matrix factorization. This hybrid similarity metric captures user–item relationships more effectively than single-measure approaches. The final prediction is generated by optimally weighting the outputs of the AKNN and SVD components, creating a weighted hybrid model. The challenge here is that the dynamic adjustment of the number of neighbours may lead to inconsistent model behaviours for users with sparse or dense user-item interactions [7].

Researchers have also turned toward clustering and multi-stage learning to capture richer similarity patterns. Sourabh et al. [16] presented a hybrid recommendation approach which uses an improved singular value decomposition applied to perform matrix factorization and a content-driven k-Nearest Neighbours model that utilized cosine similarity to identify similarities between movies based on their descriptions, year of release, and user ratings. To find the neighbours, the model used the improved kernel self-organization map with the EISEN cosine correlation distance, which helps reduce cluster overlap. Additionally, K-means clustering is used to categorize movies, where the silhouette method determines the optimal number of clusters. However, this multi-stage method increases system complexity and computational overhead during both training and prediction phases [16].

Ensemble learning has also been introduced to balance weaknesses in individual recommendation models. Ensemble learning combines multiple models either homogeneous or heterogeneous. Singh et al. [17] integrated content-based filtering, collaborative filtering, and supervised learning models with boosting algorithms. One of the best things about this work is its use of boosting to reduce individual model weaknesses. However, the system introduces added complexity due to multiple model training stages, and it does not explicitly address issues such as cold-start scenarios [17]. In a similar way Behera et al. [18] combined matrix factorization with XGBoost, feeding latent factors and contextual attributes into the boosting model. The technique captured nonlinear relationships effectively, though the computational cost remained high and cold-start challenges were not explicitly been solved [18].

Zhi-Toung et al. [19] came up with domain-specific applications by designing a hybrid recommender integrated with KNN and SVD for food recommendation, successfully uniting memory-based and model-based filtering. However, the absence of temporal and contextual features such as dietary preferences, location, or time-of-day limited its personalization capacity [19]. Explicit cold-start mitigation was focused by Juliet et al. [20] who proposed a hybrid recommendation approach to address the cold-start problem. The method integrates collaborative filtering and content-based filtering through an adaptive weighting scheme. So when rating data is sparse, the algorithm relies more heavily on content similarity; when richer in contexts, collaborative information dominates. This fusion approach outperformed traditional collaborative filtering and content filtering methods. The strength of this work lies in its explicit focus on cold-start mitigation and the use of an adaptive hybridization mechanism rather than fixed weights. The limitation is that the content-based component is simpler and does not incorporate multiple item attributes [20].

Recent advances used deep and meta-learning to improve recommendation. Liu et al. [21] proposed a hybrid model that combines a meta-learning module with an attention module to address the cold-start challenge in recommendation systems. The attention module focuses on learning personalized user interests by assigning weights to different user–item interactions. This ensures that only informative preferences have a greater contribution to the recommendation process. The meta-learning module uses Model-Agnostic Meta-Learning (MAML) to train the recommendation model in tasks, where each task corresponds to a user's preference estimation. This helps the recommendation system adapt quickly to cold-start situations. The strengths of this approach are its ability to model personalized user interests and to generalise effectively in cold-start scenarios. The disadvantages of this approach is that, the combination of attention and meta-learning increases model complexity in sparse datasets [21].

The above studies indicate that hybrid recommendation systems outperform traditional approaches by addressing sparsity and cold-start challenges. But every approach has its advantages and limitations. One research gap across the literature is the limited integration of temporal dynamics and multi-attribute content modeling, both of which are critical for capturing changing user preferences and supporting new items.

4. PROPOSED METHOD

In this section, we will first discuss our proposed model Hybrid Content and Singular Value Decomposition (HCSVD) in detail. The proposed hybrid model integrates collaborative filtering and content-based filtering to capture user preferences and item semantics effectively. This approach is better at handling cold-start problems and data sparsity by utilizing both user-item interactions and multiple content attributes. Our proposed model consists of four stages: the data pre-processing layer, the item similarity prediction layer with multiple attributes, the SVD prediction layer, and finally the hybrid prediction layer. The architecture diagram of the proposed method is shown in Fig. 3.

4.1. PRELIMINARIES

Let $U = \{u_1, u_2, ..., u_m\}$ represent the set of users, $I = \{i_1, i_2, ..., i_n\}$ represent the set of items, R be the user-item rating matrix, r_{ui} is the rating of user u for item i, and $\hat{r}_{ui}$ denote the predicted rating. Table 3. shows the notation used in our proposed approach. The goal of the proposed approach is to predict $\hat{r}_{ui}$ as accurately as possible, especially in both the normal and the cold-start settings, using the proposed hybrid model.

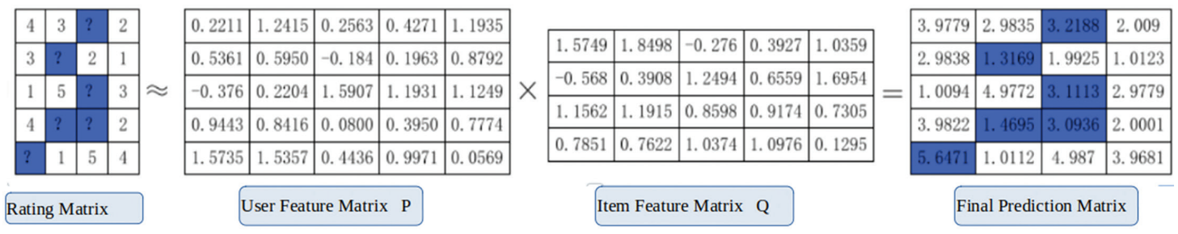

Fig. 2. SVD-based decomposition of Rating matrix R

Table 3. Notations used in the proposed hybrid system

Notation	Description
$U = \{u1, u2, \ldots, um\}$	Set of users
$I = \{i1, i2, \ldots, in\}$	Set of items
r_{ui}	Actual rating
$\mathbf{p}u, \mathbf{q}i$	Latent factor vectors
x_j	TF-IDF vector of item j (content)
$\mathbf{v}_u$	Content-based user profile vector
$\beta \in [0, 1]$	Hybrid Weight
w_j	Weight based on recency
e_{ui}	Rating prediction error
μ	Training set average rating
b_u	Bias value for user u
b_i	Bias value for item i
w_j	Weight based on recency
α	Learning rate

4.2. DATA PRE-PROCESSING

The first stage of this proposed method is data pre-processing, where the user rating matrix and item metadata are collected and extracted from the dataset. To prepare item metadata for content-based recommendation, a structured pre-processing step is applied to transform raw categorical and textual features into a format suitable for vectorization. For example, the attributes, such as item titles, category labels, and timestamp information, were extracted and cleaned. The temporal features were separated into categorical bins to capture historical trends and cold start information. Textual features such as titles were normalized by removing non-informative patterns. Categorical features, including multi-label attributes, were converted into descriptive string formats. These processed components were then concatenated to form a unified content string for each item. This text representation captures semantic, categorical, and temporal characteristics of the items. So here the metadata of the item is represented as a vector from the item metadata using an appropriate vector space model.

4.3. ITEM SIMILARITY COMPUTATION AND PREDICTION

Here the combined strings are vectorized using the TF-IDF method, resulting in a high-dimensional sparse vector. The next step is to construct a personalized user profile vector. The user profile vector is constructed by aggregating the TF-IDF vectors of the item a user has rated, weighted by the corresponding rating values. The user profile vector captures the user's preferences across multiple content dimensions.

The following equation represents the personalized user profile vector:

$$\mathbf{v}_u = \frac{1}{\sum_{j \in I_u} r_{uj}} \sum_{j \in I_u} r_{uj} \cdot \mathbf{x}_j \tag{2}$$

where V_u represent the content-based user profile vector, r_{uj} is the rating given by user u to item j. x_j is the content item vector of item j. I_u denote the set of items rated by user u.

Apart from additional user attributes, we have also included a time decay function to handle users' evolving interests as well as to get preferences for cold start items. We know that the user preference may change over time. To enhance the adaptability of the content-based recommendation system and better reflect users' evolving interests, a time-decay function is integrated into the process of constructing user profiles. Each item rated by a user contributes to the construction of their user profile based on both how much they liked it and how recent it is.

A time-aware exponential decay weight is applied to each item, defined as w_j. Hence updated user profile vector is represented as:

$$\mathbf{v}'_u = \frac{\sum\limits_{j \in I_u} w_j \cdot r_{uj} \cdot \mathbf{x}_j}{\sum\limits_{j \in I_u} w_j \cdot r_{uj}} \tag{3}$$

Where v'_u is the updated content-based user profile vector, r_{uj} is the rating given by user u to item j. x_j is the content item vector of item j. I_u denote the set of items rated by user u.

The time-aware weight based on recency w_j is calculated as:

$$w_j = \exp\left(-\lambda(t_{\text{now}} - t_j)\right) \tag{4}$$

Where λ denote decay rate, t_{now} represent the current time and t_j express the timestamp when item j interacted with the user. The exponential function exp(x) represent e^x where $e \approx 2.718$.

The content-based prediction is computed as the cosine similarity between the user's profile vector and the movie's TF-IDF vector, scaled to the original rating range. The content-based prediction equation is represented as below:

$$\hat{r}_{ui} = R_{\max} \cdot \cos(\mathbf{v}'_u, \mathbf{x}_i) \tag{5}$$

Where $\hat{r}_{ui}$ denote the predicted rating, R_{max} is the the maximum possible rating range. $\cos(v'_u, x_i)$ denote cosine similarity between the user profile vector and item feature vector.

$$\hat{r}_{ui} = R_{\max} \cdot \frac{\mathbf{v}'^{\top}_u \mathbf{x}_i}{\|\mathbf{v}'_u\| \cdot \|\mathbf{x}_i\|} \tag{6}$$

Where $\hat{r}_{ui}$ is the rating prediction, v'_u is the updated content-based user profile vector, x_i indicate the content item feature vector and R_{max} is the maximum possible rating range.

4.4. SVD BASED PREDICTION

For collaborative filtering, we are using the SVD-based MF algorithm, which models latent user and item factors learnt from historical rating data. The SVD model is used to capture latent user and item factors through matrix factorization. The changes in the user-item ratings are often influenced by user and item specific biases. We can see that a few users always give better ratings to all items, while another group may give average ratings to items. At the same time, a majority of the users provide true ratings too. To take care of these deviations, the SVD-based rating prediction formula always incorporates bias terms that represent the global average rating, individual user bias, and item bias. This adjustment helps improve the accuracy of predictions by normalizing user behaviour and item popularity.

By considering this, SVD predictive formula for ratings as:

$$\hat{r}_{ui} = \mu + b_u + b_i + \mathbf{p}_u^\top \mathbf{q}_i \tag{7}$$

Where $\hat{r}_{ui}$ is the rating prediction, μ denote the average rating, b_u and b_i implies user bias and item bias values. $\boldsymbol{p}_u$ and $\boldsymbol{q}_i$ represent the latent factor vector for user and item.

The SVD objective function for the rating prediction is represented as follows:

$$\min_{\mathbf{u},i\in D} \sum (r_{ui} - \hat{r}_{ui})^2 + \gamma \left(\|\mathbf{p}_u\|^2 + \|\mathbf{q}_i\|^2 + b_u^2 + b_i^2\right) \tag{8}$$

Where set D indicate the users and items set, γ depict regularization parameter to prevent over-fitting. $\boldsymbol{p}_u$ and $\boldsymbol{q}_i$ represent the latent factor vector for user and item, b_i and b_u implies user bias and item bias values.

The error formula for updating the bias value is as below:

$$e_{ui} = r_{ui} - \mu - b_u - b_i - \mathbf{p}_u^\top \mathbf{q}_i \tag{9}$$

Here e_{ui} indicate a difference between the expected and actual values, r_{ui} indicate the actual rating. $\boldsymbol{p}_u$ and $\boldsymbol{q}_i$ represent the latent factor vector for user and item, b_u and b_i implies user bias and item bias values.

We have used stochastic gradient descent for optimizing the result. The parameters have been updated using the stochastic gradient descent approach, as shown in equations 10 to 13.

The user bias and item bias values are updated by the equation 10 and 11 as follows:

$$\mathrm{b}_u \leftarrow b_u + \alpha(e_{ui} - \gamma b_u) \tag{10}$$

$$\mathrm{b}_i \leftarrow b_i + \alpha(e_{ui} - \gamma b_i) \tag{11}$$

The latent vectors are updated by the following equations:

$$\mathrm{p}_u \leftarrow \mathbf{p}_u + \alpha(e_{ui} \cdot \mathbf{q}_i - \gamma \mathbf{p}_u) \tag{12}$$

$$\mathrm{q}_i \leftarrow \mathbf{q}_i + \alpha(e_{ui} \cdot \mathbf{p}_u - \gamma \mathbf{q}_i) \tag{13}$$

Where α denotes the learning rate, e_{ui} indicate a difference between the expected and actual values, $\boldsymbol{p}_u$ and $\boldsymbol{q}_i$ represent the latent factor vector for user and item, b_u and b_i implies user bias and item bias values.

4.5. HYBRID PREDICTION & RECOMMENDATION

In the last stage, the final hybrid prediction is computed as a linear combination of the SVD and content-based predictions, controlled by a weighting parameter β. This fusion approach uses the strengths of content-based filtering with multiple attributes based on metadata, while SVD models latent user-item interactions from historical rating predictions. The content filtering also uses a time decay function to handle users' changing interests. Here we are using a weighting parameter $\beta\in[0,1]$ to control the contribution of each component. The final predicted rating is computed using the following equation:

$$\hat{r}_{ui}^{\mathrm{Hybrid}} = \beta \cdot \hat{r}_{ui}^{\mathrm{SVD}} + (1-\beta) \cdot \hat{r}_{ui}^{\mathrm{CE}} \tag{14}$$

where $\hat{r}_{ui}^{\mathrm{SVD}}$ is the collaborative filtering predicted score and $\hat{r}_{ui}^{\mathrm{CE}}$ is the content-based predicted score. This hybrid formulation allows the system to balance the two approaches, improving accuracy and cold-start scenarios.

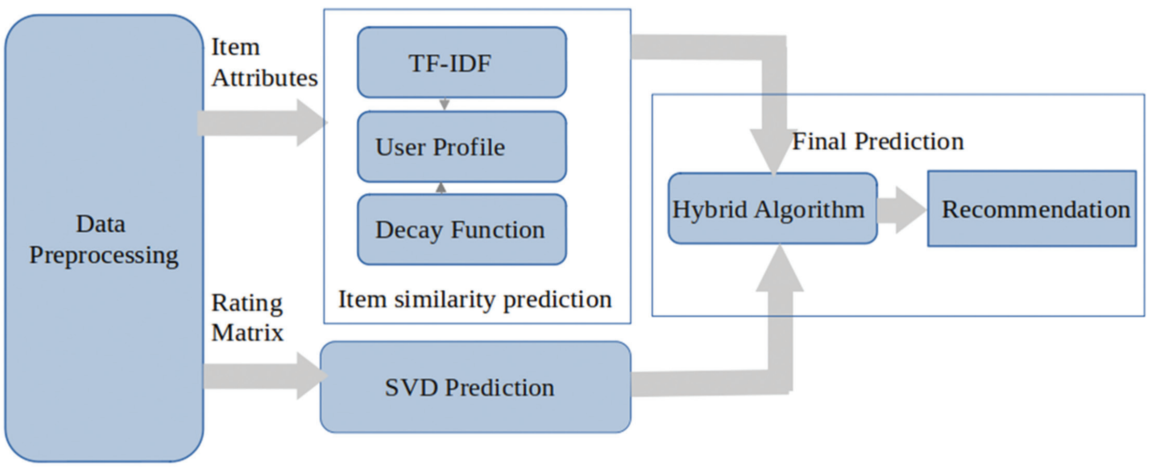

Fig. 3. Architecture diagram of HCSVD

5. EXPERIMENTS AND RESULTS

All the experiments were performed on a machine with Ubuntu 22.04 LTS powered by an 11th Gen Intel® Core™ i5-11260H CPU @ 2.60GHz×12. Python 3.9.15 is the programming language for the experimentation, and the programming environment was a Jupyter Notebook environment through Anaconda Navigator 2.4.3. The proposed method was implemented using the Scikit-Surprise library in Python [22]. Python tools like NumPy 1.23.4, Pandas 1.4.3, and Scikit-learn 1.1.1 were used for pre-processing data, evaluating metrics, and doing essential tasks. We used 75% of the dataset for training and the remaining 25% for testing. To evaluate the model's effectiveness under cold-start conditions, we conducted two controlled experiments: one for user cold-start and another for item cold-start. To test user cold start, we used users who had very few ratings, while for the item cold start case, we utilized items that had no ratings. In SVD prediction, the learning rate used is 0.01, and the number of epochs is 50. The computed time-aware weights lie in the continuous range of 0 to 1. The hybrid weight values range from 0.1 to 0.9.

5.1. DATASET

To implement the recommendation algorithms, we use two Movie-Lens datasets: ML-100K and ML-1M [23]. The Movie-Lens 100K consists of 100,000 records, where 944 users have rated 1683 items. The maximum rating given is 5, and each user has rated a minimum of 20 movies. The Movie-Lens 1M dataset consists of 1,000,209 ratings from 6,040 users on 3,952 movies. Additional metadata about the movies is available in the dataset fields, such as movie titles, release dates, and 19 types of genre vectors, with each genre represented as a binary indicator. This enables content-based methods to exploit rich categorical data. Table 4 provides a detailed description of the dataset.

5.2. BASELINE MODELS

The following baseline recommendation methods are used to compare the performance of our proposed model. They are content-based (CB) filtering, user-based KNN (UKNN) [4], item-based KNN (IKNN) [4], SVD [25], non-negative matrix factorization (NMF) [26], probabilistic matrix factorization (PMF) [27], HCFMR [18] and AMeLU [21].

CB [5, 8]: In the content-based recommendation approach, the recommendations are generated by the correlation between item attributes and the target user's profile. Each item is represented using a Vector Space Model (VSM), where features are transformed into high-dimensional vectors.

UKNN [4, 24]: This memory-based model represents users and items in a user-item rating matrix. It uses correlation-based similarity computation models, like Pearson correlation, cosine similarity, and adjusted cosine similarity, to calculate user-to-user correlations. A prediction function is then applied to generate recommendations based on these computed similarities [24].

IKNN [4, 24]: This memory-based model treats users and items as vectors in a user-item rating as a user interaction matrix. It employs correlation-based similarity computation models such as Pearson correlation, cosine similarity, and adjusted cosine similarity to determine item-to-item correlations. Recommendations are generated using a prediction function based on these calculated similarities [25].

SVD [24, 25]: Within the framework of recommendation engines, SVD can be applied to decompose the user-item interaction matrix, where users and items are represented as vectors. The resulting decomposition captures the underlying structure of the data, allowing for more accurate prediction of user preferences and generating recommendations [25].

NMF [24, 26]: Non-negative Matrix Factorization is a dimensionality reduction method that decomposes a non-negative matrix into two lower-rank matrices. NMF ensures the condition that every element within the matrices must be non-negative. In the domain of recommendation systems, NMF can be utilized to determine latent factors that represent user preferences and item traits, thereby allowing accurate rating estimates [24].

PMF [24, 27]: Recommendation employ probabilistic matrix factorization, to look for latent components that explain observed ratings by treating them as samples from a Gaussian distribution. The user-rating matrix between the user and the item in PMF is split into two lower-dimensional matrices, one for the user factors and one for the item factors [24].

HCFMR [18]: This study employs a Hybrid Collaborative Filtering with a Multi-Relation Reasoning Movie Recommendation Approach, which integrates collaborative filtering with content-based techniques. In this work, two integrated modules are used: one module learns latent user-item factors using a matrix factorization method, while another leverages item content to compute content-based similarities.

AMeLU [21]: Attentional Meta-Learned User Preference Estimator is a recommendation model for cold-start scenarios that fuses meta-learning with an attention mechanism to capture various types of user interests.

5.3. EVALUATION PARAMETERS

In our work, we utilized performance metrics mean absolute error (MAE) and root mean squared error (RMSE) to evaluate the prediction accuracy of the recommendation system [24, 28]. MAE is a popular metric for calculating the recommendation prediction. The following equation is used to compute MAE:

$$\mathrm{MAE} = \frac{\sum_{(u,i)\in T} |r_{ui} - \hat{r}_{ui}|}{|T|} \quad (15)$$

Where r_{ui} is the actual rating and $\hat{r}_{ui}$ is the predicted rating. T is the set of all user-item pairs in the test set. RMSE can be computed using the following equation:

$$\text{RMSE} = \sqrt{\frac{\sum_{(u,i)\in T} \left(r_{ui} - \hat{r}_{ui}\right)^2}{|T|}} \tag{16}$$

Where r_{ui} is the actual rating and $\hat{r}_{ui}$ is the predicted rating. T is the set of all user-item pairs in the test set.

5.4. RESULTS AND DISCUSSIONS

We have chosen the following benchmark models for our hybrid approach to evaluate performance: the content-based model CB, memory-based collaborative models UKNN and IKNN, the model-based approaches NMF, PMF and SVD, and the hybrid models HCFMR and AMeLU. Table 5 shows a performance comparison of our proposed model against the baseline models. This table shows that HCSVD achieves the lowest error rate with RMSE and MAE values of 0.8552 & 0.6745 on MovieLens 100K and 0.8451 & 0.6686 on MovieLens 1M, outperforming all baseline methods. While hybrid models such as HCFMR and AMeLU have shown better results than standalone methods, they still fall short of HCSVD. It is evident that the hybrid models outperform individual recommendation methods. We observed that the models NMF and PMF show significantly higher error rates, with NMF reaching an RMSE of 0.9671 and MAE of 0.8110 on the Movie-Lens 100K dataset. KNN-based methods, such as IKNN and UKNN, perform moderately better but still fall short of the proposed hybrid method. During the evaluation process, we observed that CB and IKNN individually display nearly identical error values. The SVD model demonstrated superior performance in capturing latent user–item interactions, achieving lower prediction errors compared to other benchmark models. However, hybrid models gave better results by combining both collaborative and content-based features.

We have conducted additional experiments by varying the number of recommendations to evaluate how our method performs compared to other approaches in terms of prediction accuracy. As shown in Fig. 4 and Fig. 5, our approach HCSVD performed better than all other methods across different values of number of N={10,20,30,50,80}, where N is the number of recommendations. However, we observed that prediction accuracy gradually dropped as the number of recommendations increased.

Table 4. Movie-Lens Dataset Details

Dataset	No of Users	No of Items	Total no of Ratings	Density(%)
ML-IOOK	944	1683	100000	6.37
ML-IM	6040	3706	1000209	4.47

Table 5. Analysis of the Proposed method with Baseline models

	Move-Lens 100K		Move-Lens 1M	
Model	**RMSE**	**MAE**	**RMSE**	**MAE**
NMF [27, 25]	0.9671	0.8110	0.9213	0.7986
PMF [28, 25]	0.9590	0.7980	0.9108	0.7656
CB [5, 8]	0.9571	0.7610	0.9375	0.7263
IKNN [4, 25]	0.9500	0.7631	0.9118	0.7385
UKNN [4, 25]	0.9454	0.7435	0.9107	0.7185
SVD [23, 24]	0.9076	0.7146	0.9013	0.7087
HCFMR [19]	0.8850	0.6970	0.8210	0.6291
AmeLU [21]	0.8822	0.7277	0.8756	0.7068
HC SVD	**0.8552**	**0.6745**	**0.8451**	**0.6686**

We have also tested the performance of our proposed recommendation algorithm for cold-start users/items. Here we have conducted it in two ways: 1. Measure how well the model predicts for users with few or no historical ratings; 2. Measure performance on items with no prior ratings in training. In the first case, evaluation is performed, focusing on users with limited interaction history. In this setting, cold-start users are selected by identifying those with very few ratings for training. The remaining ratings for these users are held out for testing. In the second scenario, we have evaluated the model's ability to handle new items. In this evaluation, a subset of items around 5% is randomly selected without rating and removed from the training set. These items are then included only in the test set.

To evaluate the performance of the proposed hybrid model HCSVD under item cold-start conditions, we conducted a series of experiments by adjusting the hybrid prediction parameter β. The impact of hybridization parameter β is directly applied to SVD prediction and 1-β applied to the content prediction. Tables 6 and 7 present the results for the cold-start user scenario, while Tables 8 and 9 present the results for the cold-start item scenario.

Table 6. HCSVD (Cold-Start Users) on Movie Lens 100K

Beta (β)	0.9	0.7	0.5	0.3
MAE	0.7953	0.7498	0.7212	0.6987
RMSE	0.9948	0.9412	0.8911	0.8742

Table 7. HCSVD (Cold-Start Users) on Movie Lens 1M

Beta (β)	0.9	0.7	0.5	0.3
MAE	0.7845	0.7552	0.7208	0.6948
RMSE	0.9875	0.9644	0.8669	0.8711

Table 8. HCSVD (Cold-Start Items) on Movie Lens 100K

Beta (β)	0.9	0.7	0.5	0.3
MAE	0.7934	0.7326	0.7154	0.6939
RMSE	0.9820	0.9027	0.8842	0.8625

Table 9. HCSVD (Cold-Start Items) on Movie Lens 1M

Beta (β)	0.9	0.7	0.5	0.3
MAE	0.7710	0.7578	0.7275	0.6892
RMSE	0.9175	0.8641	0.8577	0.8469

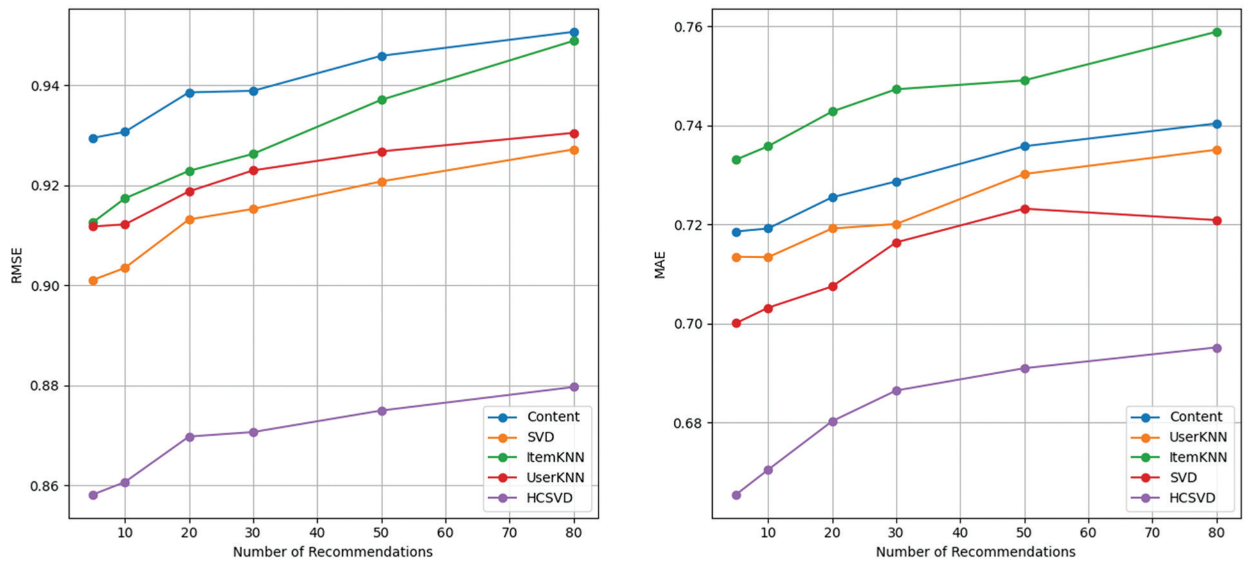

Fig. 4. Evaluation of RMSE and MAE with Varying Number of Recommendations Across Models on Movie Lens 100K

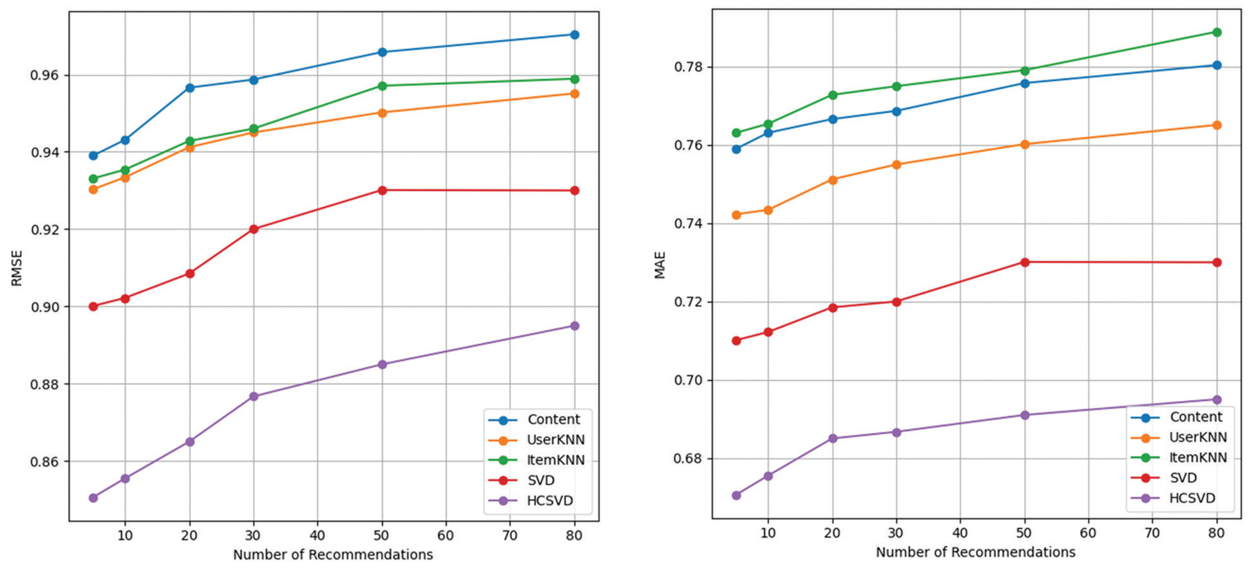

Fig. 5. Evaluation of RMSE and MAE with Varying Number of Recommendations Across Models on Movie Lens 1M

Considering the cold-start item performance, when $\beta = 0.5$ where content-based filtering and the SVD model contribute equally the system achieves an MAE of 0.7275 and RMSE of 0.8577. This demonstrates that the model maintains stable and balanced performance when both components are equally weighted. When $\beta = 0.3$, where content-based filtering contributes 70%, the model produces lowest MAE (0.6892) and RMSE (0.8469). This performance is remarkably close to the normal (non-cold-start) case, where the MAE and RMSE were 0.6686 and 0.8469, respectively. These findings are summarized in Table 9.

The fusion of singular value decomposition enables the model to capture unknown patterns in the user-item matrix. At the same time, incorporating multiple item attributes through content-based filtering enhances prediction accuracy. The hybrid model HCSVD outperforms other baselines in this setting due to its ability to depend on the content-based.

6. CONCLUSION

Currently, the recommendation of a cold-start issue remains an open subject matter, and the recommendation system continues to face a significant challenge when formulating this recommendation. In this paper, we propose a hybrid recommendation model HCSVD that combines content-based filtering and collaborative filtering to address challenges in recommendation systems, particularly cold-start scenarios. Our method utilizes the vectorization of multiple item features, as the content-based component was able to make meaningful predictions even in the scarcity of historical user-item interactions. The method uses a time-aware exponential decay function derived from the item's timestamp feature to construct the user profile. This approach places greater emphasis on more recently rated items, thereby enhancing the relevance of the user's preference context.

Compared to benchmark models, our proposed method achieves a 3.06% reduction in RMSE and a 3.23% reduction in MAE, demonstrating its superiority in prediction accuracy. Experimental results indicate HCSVD has better performance in prediction accuracy over other benchmark models in normal and cold-start situations. In the future, we are planning to enhance our methods by integrating deep learning techniques and other innovative data representations like knowledge graphs for better recommendations. In future work, we also plan to extend the system for both rating prediction and ranking recommendations. Using both prediction and ranking evaluations will help the system to measure user satisfaction and usefulness more effectively.

7. REFERENCES:

[1] Y. Sun, Q. Liu, "Collaborative filtering recommendation based on k-nearest neighbor and non-negative matrix factorization algorithm", The Journal of Supercomputing, Vol. 81, 2024, p. 79.

[2] L. Wu, P. Sun, R. Hong, Y. Ge, M. Wang, "Collaborative neural social recommendation", IEEE Transactions on Systems, Man, and Cybernetics: Systems, Vol. 51, 2021, pp. 464-476.

[3] R. Duan, C. Jiang, H. K. Jain, "Combining review-based collaborative filtering and matrix factorization: A solution to rating's sparsity problem", Decision Support Systems, Vol. 156, 2022, p. 113748.

[4] R. Chen, Q. Hua, Y. S. Chang, B. Wang, L. Zhang, X. Kong, "A survey of collaborative filtering-based recommender systems: From traditional methods to hybrid methods based on social networks", IEEE Access, Vol. 6, 2018, pp. 76292-76326.

[5] P. Lops, M. de Gemmis, G. Semeraro, "Content-based recommender systems: State of the art and trends", Recommender Systems Handbook, Springer, 2011.

[6] Y. Koren, R. Bell, C. Volinsky, "Matrix factorization techniques for recommender systems", Computer, Vol. 42, No. 8, 2009, pp. 30-37.

[7] T. Roy, P. Shetty, "A Hybrid Approach to Predict Ratings for Book Recommendation System using Machine Learning Techniques", Proceedings of the IEEE Region 10 Symposium, New Delhi, India, 27-29 September 2024, pp. 1-6.

[8] F. Ricci, L. Rokach, B. Shapira, "Rating singular value decomposition: An enhanced matrix factorization technique for recommender systems", Recommender Systems Handbook, Springer, 2015, pp. 291-324.

[9] T. Widiyaningtyas, M. I. Ardiansyah, T. B. Adji, "Recommendation algorithm using SVD and weight point rank (SVD-WPR)", Procedia Computer Science, Vol. 161, 2019, pp. 849-856.

[10] M. J. Pazzani, D. Billsus, "Content-based recommendation systems", The Adaptive Web, Lecture Notes in Computer Science, Vol. 4321, Springer, 2007.

[11] G. Salton, M. J. McGill, "Introduction to Modern Information Retrieval", McGraw-Hill, 1983.

[12] F. Ricci, L. Rokach, B. Shapira, P. B. Kantor, "Recommender systems handbook", Springer, 2015.

[13] S. Lv, J. Wang, F. Deng, Y. Li, Y. Zhang, "A hybrid recommendation algorithm based on user nearest neighbor model", Scientific Reports, Vol. 14, 2024, p. 17119.

[14] J. Guan, B. Chen, S. Yu, "A hybrid similarity model for mitigating the cold-start problem of collaborative filtering in sparse data", Expert Systems with Applications, Vol. 242, 2024, p. 123700.

[15] A. Liang, Y. Bai, M. Wu, J. Wu, G. Wu, "Research on personalized recommendation algorithms for different user behaviors", Proceedings of the 4th International Conference on Big Data, Artificial Intelligence and Risk Management, Guangdong China, 15-17 November 2024, pp. 163-169.

[16] S. Sharma, H. K. Shakya, "Hybrid recommendation system for movies using artificial neural network", Expert Systems with Applications, Vol. 258, 2024, p. 125194.

[17] K. Singh, S. Dhawan, N. Bali, "An ensemble learning hybrid recommendation system using content-based, collaborative filtering, supervised learning and boosting algorithms", International Journal of Electrical and Computer Engineering, Vol. 13, No. 5, 2023, pp. 5599-5608.

[18] G. Behera, S. K. Panda, M.-Y. Hsieh, K.-C. Li, "Hybrid collaborative filtering using matrix factorization and XGBoost for movie recommendation", Electronics, Vol. 13, No. 8, 2024, p. 1490.

[19] Z.-T. Yap, S.-C. Haw, N. Ruslan, "Hybrid-based food recommender system utilizing KNN and SVD approaches", Cogent Engineering, Vol. 11, No. 1, 2024, p. 2436125.

[20] A. N. M. Juliet, "An improved hybrid recommendation system algorithm for resolving the cold-start issues", Journal of Information Systems Engineering & Management, Vol. 10, No. 2, 2025, pp. 243-250.

[21] S. Liu, Y. Liu, X. Zhang, C. Xu, J. He, Y. Qi, "Improving the performance of cold-start recommendation by fusion of attention network and meta-learning", Applied Sciences, Vol. 13, No. 2, 2023, p. 1120.

[22] N. Hug, "Surprise: A Python library for recommender systems", Journal of Open Source Software, Vol. 5, 2020, p. 2174.

[23] F. M. Harper, J. A. Konstan, "The MovieLens Datasets: History and Context", ACM Transactions on Interactive Intelligent Systems, Vol. 5, No. 4, 2015, pp. 1-19.

[24] C. Wenga, M. Fansi, S. Chabrier, J.-M. Mari, A. Gabillon, "A comprehensive review on non-neural networks collaborative filtering recommendation systems", Journal of Machine Learning Theory, Applications and Practice, Vol. 1, No. 1, 2023, pp. 1-44.

[25] X. Zhou, J. He, G. Huang, Y. Zhang, "SVD-based incremental approaches for recommender systems", Journal of Computer and System Sciences, Vol. 81, 2015, pp. 717-733.

[26] X. Zhang, X. Zhou, L. Chen, "Explainable recommendations with nonnegative matrix factorization", Artificial Intelligence Review, Vol. 56, 2023, pp. 3927-3955.

[27] A. Mnih, R. Salakhutdinov, "Probabilistic matrix factorization", Advances in Neural Information Processing Systems, Curran Associates, Inc., 2008, pp. 1257-1264.

[28] W. Zhu, "Statistical parameters for assessing environmental model performance related to sample size: Case study in ocean color remote sensing", Remote Sensing of Environment, Vol. 280, 2022, p. 13179.