Indonesian Journal of Electrical Engineering and Informatics (IJEEI)
Vol. 13, No. 4, December 2025, pp. 897~910
ISSN: 2089-3272, DOI: 10.52549/ijeei.v13i4.6780

Enhancing Recommendation Systems Through a Hybrid Fuzzy-Sparse Similarity Model and Optimizing Noise Reduction

Kausar Salim Attar[1], Ashish Jadhav[2]
[1]PhD Scholar, Department of Computer Engineering, Ramrao Adik Institute of Technology, Navi Mumbai, India.
[2]Professor, Department of Information Technology, Ramrao Adik Institute of Technology, Navi Mumbai, India.

Article Info

Article history:

Received Jun 8, 2025
Revised Oct 29, 2025
Accepted Nov 13, 2025

Keyword:

Recommender system,
Fuzzy set,
Data Sparsity,
Similarity measures.

ABSTRACT

The vast amount of information available on the Internet and e-commerce has led to the development of recommendation systems that help users find relevant products or content. As digital applications continue to grow worldwide, ensuring the right user experience in a short period of time remains a significant challenge. With the increasing use of mobile devices, ensuring access to accurate and timely information has become an essential part of today's business operations. The accuracy of the estimates depends not only on the methodology used, but also on the accuracy of the data. External factors and unexpected noise issues can affect users in the rating process. This problematic influence, coming from well-intentioned users which can lead to distortion of the rating results during the recommendation process. In this study, we present a Hybrid Fuzzy-Sparse Similarity (HFSS) methodology designed to improve the accuracy of recommendation and reduce the error caused by source noise due to rating sparsity. Initially, a limited amount of data utilized in a form of rating matrix along with the sparse distribution, is collected for the analysis of the recommendation process. An extended fuzzy set matrix creation mechanism is proposed to solve the existing sparsity problems. By, using the extended sparse matrix, the similarity of models is calculated from the complex set theory, which allows for model-based recommendations. The proposed HFSS model is evaluated on the Movie lens database. First, the system recommendation performance evaluation is made and later a comparison performance is measured by MAE, RMSE, and F1 score metrics, which demonstrates better recommendation accuracy and performance than the comparing performance-based methods.

Copyright © 2025 Institute of Advanced Engineering and Science.
All rights reserved.

Corresponding Author:

Kausar Salim Attar,
Department of Computer Engineering,
Ramrao Adik Institute of Technology,
Navi Mumbai, India.
Email: kausarattar7@gmail.com

1. INTRODUCTION

The proliferation of online information and the growth of e-commerce have created a problem known as information overload, which makes it difficult for users to find products that interest them. To address this problem, recommender systems (RS) have been developed to help users identify their needs and find the products they need [1]. These tools have been used in many areas, such as knowledge-based services, eco-friendly hotel selection, and classification [2], [3]. Well-designed recommendation systems [4], [5] can provide personalized services and create a stronger connection between users and the system. Studies have shown that these factors influence consumer decisions and contribute to e-commerce. However, there are few studies that examine how levels of knowledge affect the results of recommendations. Studies [6] and [7] have shown that the information used in recommendation systems can be subject to noise, which can affect the performance of the system due to expected and unexpected events.

Mostly RS relies on user and product information, the underlying algorithm, and the resulting recommendations. These systems are typically divided into three main categories: content-based RS [8], collaborative filtering RS [9], and hybrid RS [10]. In recommendation systems, it has been observed that while many users engage with the platform, they often fail to provide meaningful feedback or ratings for the products [11]. To reduce the impact of natural noise, several additional measures are suggested to reassess the rating values. In [12], RS requests users to enter the same value multiple times at different intervals. In [13] used an interactive system to guide accuracy over time, thereby improving the accuracy of historical expert estimates. While these methods are effective in natural noise, they require a lot of work and are too complex for large-scale RS implementations.

Incompleteness occurs when the user's satisfaction with the product is low due to sparsity issue. In other words, the matrix is incomplete, leading to sparseness issues, a common problem in RSs [14], [15]. This problem is particularly important in collaborative filtering (CF) projects, where incomplete information leads to difficulties in making accurate decisions. As a result, this problem presents a great opportunity for researchers to explore new ways to estimate missing values in the matrix [16]. The two primary categories of these methods are classification-based and fuzzy-based. The classification method [17] divides the scores into three categories: low, medium, and high, with the boundaries of the categories. However, this method divides the dimensions into equal parts, which does not capture the specificity of the dimensions. Fuzzy feature-based methods [18], on the other hand, use complex logic to assign scores to groups with different levels of significance, resulting in a more consistent representation than classification-based methods. Fuzzy sets [19] provide solutions by capturing the differences in levels, which helps to calculate the similarity of the system even if the levels are not the same.

This paper is structured as follows. Section 2 mainly analyzes the related works. Section 3 introduces the Hybrid Fuzzy-Sparse Similarity mechanism for the recommendation system. Section 4 evaluates the proposed works and provides a result analysis. Section 5 provides the conclusions and prospects of the study.

2. RELATED WORKS

Existing research on recommender systems has focused on improving recommendation accuracy, handling data sparsity, and managing uncertainty in user ratings. Traditional collaborative filtering (CF) and content-based filtering (CBF) remain foundational approaches. However, CF often suffers from sparse rating matrices and noise in user feedback, while CBF relies heavily on detailed item attributes. To address these issues, hybrid and fuzzy-based models have been proposed.

Yera et al. (2020, 2024) introduced fuzzy-based methods to manage natural noise and inconsistencies in ratings by applying fuzzy membership functions to user profiles. Baek et al. (2023) normalized noisy ratings before computing similarity to enhance reliability. Similarly, Wang et al. (2021) developed a fuzzy-based noise management approach for collaborative filtering, demonstrating improved stability in uncertain data. Recent hybrid fuzzy-sparse models combine fuzzy logic with matrix factorization to reduce sparsity and model uncertainty simultaneously.

Despite these advances, few studies explicitly integrate fuzzy-set representation with sparse-matrix extension in similarity computation. The proposed HFSS framework contributes by combining fuzzy logic to capture uncertainty and sparse similarity mechanisms to maintain recommendation accuracy even with limited ratings.

Yera et al. [20] presented a tool designed to recommend products based on users' preferences and preferences, which has been widely used in fields such as e-commerce, e-learning, and social networks. These systems rely on understanding user preferences that are often inaccurate due to factors such as human error or subtle human behaviour. These issues, called natural noise, can affect the recommendations and affect the performance of the system. To address these challenges, this chapter proposes two fuzzy models to manage natural noise, allowing a more robust and robust representation of uncertainties in user profiles. These models serve as a data processing step before generating recommendations, focusing on individual and group recommendation scenarios. The case studies demonstrate that the proposed method facilitates the performance of individual and group RS.

R. Yera et al. [21] proposed a multi-step approach that includes pattern detection, filtering, noise level analysis, and then identification and correction of noisy data. Their approach uses fuzzy logic to deal with uncertainty, and they emphasize the importance of regularities in ratings to detect and reduce biases. Collaborative filtering (CF) is a widely used recommendation technique in academic research and real-world applications [22]. It identifies the top N users or an element based on similarity measures and maximizes the preferences of the selected neighbours to generate recommendations.

Baek et al. [23] proposed a method to address the challenges of generating recommendations from noisy and sparse datasets. They emphasized that noisy ratings should be normalized before performing tasks

such as constructing a similarity matrix or predicting ratings. Gathering input data is an essential step in any recommendation system. Input data typically fall into two categories: implicit exploration data and explicit user ratings. Fuzzy data includes analysis of users' browsing behaviors, such as time spent on pages, navigation behaviors, and actions [24].

To solve this problem, researchers have explored ways to improve the matrix density. For example, Zhang et al. [16] solved the cold start problem by adopting a randomized algorithm suitable for small data sets and then adopting a hybrid recommendation strategy to simplify the optimization process easily as more data becomes available. Similarly, Li et al. [25] proposed a hybrid model combining content-based filtering and CF. Increasing user engagement and rating input in CF helps fill in the user-item matrix, resulting in better recommendations.

Javed et al. [26] proposed a system that provides personalized recommendations using metadata associated with content. Content-based RS analyzes metadata such as titles, story details, and visual content to identify and recommend similar content that the user has previously engaged with. These systems are easy to use, but their performance is limited by their dependence on the user's previous actions, which makes them less efficient than CF methods.

Wang et al. [27] proposed a recommendation framework based on existing relationships to improve sparse matrix factorization using low-rank structures. This method determines the user's preferences for features by mixing smaller quantities of features. This model effectively addresses challenges such as data sparsity and cold start issues by integrating previous assumptions. Sahu et al. [28] proposed a comprehensive robustness assessment method for web applications by combining a hesitant fuzzy-based symmetric method with the analytic hierarchy process (AHP) and the Technique for Order of Preference by Similarity to the Ideal Solution (TOPSIS) in the same manner as the ideal solution for assessing the durability of web applications.

Fu et al. [29] proposed a deep learning-based model for CF that captures interactions between users and features to generate recommendations for others. This method uses deep learning techniques instead of the traditional cosine similarity algorithm, providing greater flexibility and enhancing the role of matrix factorization in data selection and normalization. Despite these advantages, the model struggles to deal with the cold start problem, especially when there is insufficient information about new features.

F. Wang et al. [30] proposed a comprehensive CF recommendation framework that combines base rating predictions with user, item, and trust data. By integrating the reliability assessment into the existing CF method, the method effectively solves the cold start problem by augmenting the evaluation information with the reliability assessment. Zhang et al. [33] introduced a new recommendation method called Depth Variational Matrix Factorization (DVMF) which is suitable for handling large and small data sets.

N. Aamer et al. [34] proposed AI-driven e-commerce recommenders use collaborative filtering—especially UBCF—enhanced with preprocessing, normalization, and PCA to improve accuracy, scalability, and personalized preference prediction. Studies show that combining SMOTE for data balancing, PCA for dimensionality reduction, and LSTM for classification significantly improves social media sentiment prediction by handling imbalance, reducing redundant features, and capturing key context [35]. Opinion mining extracts product details and classifies reviews as positive or negative, enabling efficient summaries and helping buyers make informed decisions [36].

3. PROPOSED HFSS METHODOLOGY

Similarity between products or users can be analyzed using a rating matrix. In a fuzzy set framework, both concepts align fundamentally. When a product is represented as an element of the fuzzy set, the focus shifts to measuring product similarity, leading to the development of a product-based CF recommendation algorithm. On the other hand, considering users as objects, we can construct user-based CF algorithms by determining user similarities. The proposed hybrid fuzzy-sparse similarity (HFSS) model captures the measurement uncertainty using fuzzy sets and constructs a similarity method that enhances the recommendations product by using sparse product matrices as discussed below.

3.1. Fuzzy Set (FS) Mechanism for Rating Uncertainty

Torra [29] introduced the concept of hesitant FS, which is an extension of traditional FS, to capture uncertainties. This reflects the situation in which a person is in doubt between several potential membership values for an object in a set. Listing these various values can help to accurately reflect the uncertainty inherent in decision-makers' judgments and judgments. Xu et al.[31] describes the fuzzy mechanism using hesitant FS as follows.

Let Z denote a reference set and M denote a collection of membership functions $f_m(z)$, each of which is stored in Z and takes values in the range 0 to 1 for a fuzzy set (FS) . This can be expressed as:

$$M = \{< z, f_M(z) > | z \in Z\}$$

The set $f_z = \{\beta | \beta \in f_M(z)\}$ contains unique values within the interval [0, 1], where β indicates the degree to which $z \in Z$ belongs to M. The function $f_m(z)$, referred to as a fuzzy element, serves as a core component of a fuzzy set (FS).

For instance, if $Z = \{z_1, z_2, z_3\}$ is a predefined set, the FS membership values for each z_i for M. Let $Z=\{Movie_1, Movie_2, Movie_3\}$ and the ratings (membership values) for each movie in M are as follows:

$f_m(Movie_1) = \{0.8,0.7,0.9\}$: Viewer ratings for ***Movie₁*** are 0.8, 0.7, and 0.9.
$f_m(Movie_2) = \{0.6,0.7\}$: Viewer ratings for ***Movie₂*** are 0.6 and 0.7.
$f_m(Movie_3) = \{0.5,0.4,0.6,0.7\}$: Viewer ratings for ***Movie₃*** are 0.5, 0.4, 0.6, and 0.7.

The fuzzy set M can then be expressed as, $M = \{<Movie_1,\{0.8,0.7,0.9\}>, <Movie_2,\{0.6,0.7\}>, <Movie_3,\{0.5,0.4,0.6,0.7\}>\}$. Here, each β value in $f_m(z)$ reflects a viewer's degree of satisfaction with the corresponding movie.

In FS theory, the membership function represents not a single value but a range of values, capturing uncertainty or uncertainty. In e-commerce recommendation systems, numerous users provide various ratings, each reflecting a degree of uncertainty. Thus, the rating of a product can be seen as a collection of user evaluations, forming a set of fuzzy numbers that collectively create a rating fuzzy set. This perspective allows the interpretation of diverse user ratings for a product as expressions of uncertainty regarding that Product.

While the equations in the proposed HFSS framework appear numerically defined, the rating representation and similarity computation are inherently fuzzy in nature. Each rating value is transformed into a **membership degree (β)** within the interval [0, 1], representing the degree of satisfaction or uncertainty rather than a fixed score. All subsequent computations—such as distance and similarity measures—are thus performed over fuzzy membership functions, making the process non-deterministic and capable of handling uncertainty in user evaluations.

3.2. Similarity Computation

Similarity measures play a crucial role across numerous scientific domains, such as decision-making, pattern recognition, machine learning, and market forecasting. Considerable research has been dedicated to this topic, particularly in the context of FS. Xu et al. [32] pioneered the development of various distance measures for fuzzy sets, deriving these from their proposed similarity metrics.

Consider two fuzzy sets, M_1 and M_2, defined over the domain Z. The distance between these sets, denoted as $dist(M_1, M_2)$, satisfies three key properties. First, the value of the distance is constrained within the range [0, 1], where 0 represents no separation and 1 represents maximum separation. Second, the distance equals zero only when the two sets are completely identical. Third, the distance is symmetric, meaning the order of comparison does not matter; that is, $dist(M_1, M_2)$ is always equal to $dist(M_2, M_1)$.

Let's consider a scenario involving product ratings. Suppose there are two products, A and B, each rated by customers on a scale of 0 to 5. The ratings can be represented as fuzzy sets M_1 and M_2 over the universe $Z=\{0,1,2,3,4,5\}$, where the membership degree indicates how strongly a rating applies to each score.

For product A the $M_1 = \{(0,0), (1,0.1), (2,0.3), (3,0.5), (4,0.8), (5,1)\}$, and for product B the $M_2 = \{(0,0), (1,0.2), (2,0.4), (3,0.6), (4,0.9), (5,1)\}$. Now compute the distance $dist(M_1, M_2)$ using a standard fuzzy set distance measure, such as the Manhattan distance normalized using Eq. (1) to fall between 0 and 1:

$$dist(M_1, M_2) = \frac{1}{n}\sum_{z \in Z} |M_1(z) - M_2(z)| \quad (1)$$

Here, n is the number of elements in M. If the sets were identical, the distance would be 0. Conversely, a distance closer to 1 would imply significant differences in their ratings.

For two fuzzy sets, M_1 and M, defined on a universal set Z, their similarity measure, denoted as $sim(M_1, M_2)$, must satisfy three key properties. First, the similarity value should always fall within the interval of 0 and 1. Second, the measure reaches a value of 1 exclusively when M_1 and M_2 are the same. Third, the similarity measure must be symmetric, meaning the relationship is consistent in both directions, such that $sim(M_1,M_2)$ is equal to $sim(M_2,M_1)$.

Let's consider an example where fuzzy sets M_1 and M_2 represent customer satisfaction ratings for two products on a scale of 0 to 10. The universal set Z is $\{0, 1, 2, ..., 10\}$. Each fuzzy set assigns a membership value to elements of Z, reflecting how well each rating describes the satisfaction level.

Satisfaction ratings for product *A* membership values M_1 = {(*6,0.8*), (*7,1.0*),(*8,0.9*), (*9,0.6*)}. This indicates customers primarily rate product *A* between 6 and 9, with the highest satisfaction at 7, similarly for Product *B* membership values M_2 = {(*6,0.7*), (*7,1.0*), (*8,0.8*), (*9,0.5*)}. This shows a similar trend, with peak satisfaction also at 7 but slightly lower values for other ratings. Calculating the similarity $sim(M_1,M_2)$ using a common method like the cosine similarity will provide the similarity measure. If it is very close to *1* then it indicates the satisfaction ratings for Product *A* and Product *B* are nearly identical, else if 0 then un-identical.

Calculating the similarity between FS requires that both sets contain an equal number of elements. However, this condition is often challenging to meet in practice due to the inherent diversity of FS. Therefore, it becomes necessary to address this disparity by adding elements to the sets before performing the similarity computation. Various methods, such as using the average value or modal number, can be employed for this purpose. However, ensuring the effectiveness and accuracy of these complementary techniques remains a critical challenge in this context.

The similarity $sim(M_1,M_2)$ between two FS is calculated as $1- dist(M_1, M_2)$, where $dist(M_1, M_2)$ represents their distance. A smaller distance implies higher similarity. For fuzzy values, this similarity depends on the distance between their membership functions, similar to how it is measured for FS. Xu et al. [33] proposed a formula to determine the similarity between two fuzzy values f_1 and f_2 using distance metrics as given in Eq. (2).

Additionally, the proposed similarity measure between fuzzy sets satisfies the triangle inequality, ensuring metric consistency and confirming that the indirect similarity between two fuzzy sets through a third set cannot exceed their direct similarity. This property reinforces the theoretical soundness of the HFSS similarity computation.

$$sim(f_1, f_2) = \frac{\sum_{i=1}^{l} f_1(i) \cdot f_2(i)}{\sqrt{\sum_{i=1}^{l}(f_1(i))^2 \cdot \sum_{i=1}^{l}(f_2(i))^2)}} \quad (2)$$

where, $f_1(i)$ and $f_2(i)$ denotes the *i*-th largest elements in f_1 and f_2 respectively. The similarity measure for FS values is based on the concept of Pearson's similarity measure. This approach efficiently minimizes the uncertainty that can arise when comparing two FS values.

Let's consider two fuzzy sets M_1 and M_2 representing user ratings for two movies across a set of criteria such as *acting*, *storyline*, and *cinematography*. Each criterion is evaluated on a scale from 0 to 1, where 1 indicates the highest satisfaction.

Suppose M_1 (M_A ratings) and M_2 (M_B ratings) have the fuzzy membership values for ***Acting*** is M_1 =0.9, M_2=0.8; *Storyline* is M_1=0.7, M_2=0.9; *Cinematography* is M_1=0.8, M_2=0.85. Using Eq. (2) we compute $sim(M_1,M_2)$, where l=3 (three criteria).

$\sum_{i=1}^{3} M_1(i) \cdot M_2(i)$ = 2.03, and $\sqrt{\sum_{i=1}^{3}(M_1(i))^2 \cdot \sum_{i=1}^{3}(M_2(i))^2)}$ = 2.052

So, $sim(M_1, M_2) = \frac{2.052}{2.03} \approx 0.989$

The similarity score between the ratings of M_A and M_B is approximately 0.989, indicating they are highly similar in terms of the given criteria.

3.3. User and Product Rating for Recommendation using HFSS

A. User Rating Mechanism

In an e-commerce recommendation system, user ratings can vary significantly based on individual personality traits. For example, some individuals may view a four-star rating as very positive, while others might see it as inadequate. To address these inconsistencies and reduce uncertainty, it is important to normalize user ratings before further processing.

In a recommendation system, the rating R_{up} represents the score given by a user *u* to a product *p*. The average rating $\bar{R}_u$ for a user is calculated by dividing the total sum of ratings $\sum_{p=1}^{t} R_{up}$ with the number of products *t* that the user has rated. To measure the satisfaction level for a specific rating, the membership degree β_{up} is determined using Eq. (3).

$$\beta_{up} = \frac{R_{up}-R_{min}}{R_{max}-R_{min}}, \quad \beta_{up} \in [0,1] \quad (3)$$

Here, R_{min} and R_{max} are the minimum and maximum possible ratings, respectively, and β_{up} quantifies satisfaction on a normalized scale between 0 and 1.

Let's consider the rating scale ranges from R_{min} = 1 to R_{max} = 5. User u rates three products as Product1 as $R_{u1}=4$, Product2 as $R_{u2}=5$, and Product3 as $R_{u3}=5$. The total number of products rated by the user, t=3. The average rating $\bar{R}_u = \frac{R_{u1}+R_{u2}+R_{u3}}{t}$ = 4, and the membership degree (β_{up}) for each product as $\beta_{u1}=0.75$, $\beta_{u2}=1.0$, and $\beta_{u3}=0.5$. This makes ratings normalized into a satisfaction level, which makes it easier to compare different ratings on a common scale.

Rating scales commonly use a range like 0 to 5, where the maximum rating (R_{max}) is 5 and the minimum (R_{min}) is 0. On a five-point scale, the membership degree can be easily calculated as $\beta_{up} = \frac{R_{up}}{5}$. Other scales, such as three-point or ten-point systems, are also popular, particularly for assessing aspects like after-sales service quality. A user's rating provides a measurable indication of their satisfaction level, reflecting the extent of their preference or approval. This approach ties the membership degree directly to the user's personal experience and perception.

B. Product Rating Mechanism

The ratings assigned to P_p within the system can be represented as a fuzzy set (FS) as, $f(\mathrm{P_p}) = \{\beta_{up}, \beta_{2p}, \ldots, \beta_{l_p p}\}$, where l_p represents the total number of ratings that P_p receives.

In CF recommendations, suggesting products is redefined as recommending FS, where product similarity translates into FS similarity. However, a challenge arises when the FS derived from a rating matrix has differing sizes, denoted as $l_p \neq l_q$. To compute similarity effectively, FS must be of equal size. Addressing this discrepancy requires extending smaller FS to match the largest set size among all considered sets $\{l_1, l_2, \ldots, l_p, l_q, \ldots\}$.

To achieve this, the elements of $f(P_p)$ and $f(P_q)$ are arranged in ascending order. Two fuzzy sets are defined as equivalent, $f(P_p) = f(P_q)$, if and only if all corresponding elements β_{up} and β_{uq} satisfy the condition $\beta_{up} = \beta_{uq}$ for u=1,2,…,l, where l is the number of elements in the extended sets. If a set contains fewer elements, additional elements are incrementally appended to it until its size equals the maximum l_{max} among all sets. This mechanism ensures consistent representation and enables meaningful comparison by enforcing uniformity in the sizes of the FS.

To formalize the process of equalizing the sizes of fuzzy sets $f(P_p)$ and $f(P_q)$, we define an extended fuzzy set $f^*(P_p)$ for each product. Let l_p and l_q represent the original sizes of the two sets, and l_{max}= max(l_p , l_q) denotes the maximum size. The extension process is represented as:

$$f^*(P_p) = \begin{cases} f(P_p), & if\ l_p = l_{max}, \\ f(P_p) \cup \{\gamma 1, \gamma 2, \ldots, \gamma l_{max} - l_p\}, & if\ l_p < l_{max}, \end{cases}$$

where $\{\gamma 1, \gamma 2, \ldots, \gamma l_{max} - l_p\}$ are additional elements added to $f(P_p)$ to reach the size l_{max}. A similar extension applies to $f(P_q)$. The equivalence condition is then formulated as:

$$f^*(P_p) = f^*(P_q) \Leftrightarrow \beta_{up} = \beta_{uq}, \quad \forall u = 1,2, \ldots, l_{max}$$

where β_{up} and β_{uq} represent the u-th elements of the sorted extended sets $f^*(P_p)$ and $f^*(P_q)$, respectively.

In a personalized recommendation system, the objective is to suggest a product p to a user u by incorporating their preferences. The addition of additional elements to the FS is guided by user u's historical rating tendencies, following two distinct conditions based on the user's average rating ($\bar{R}_u$).

***Condition 1*:** If $\bar{R}_u \leq 4$, it indicates that user u tends to assign lower ratings overall. To adjust for this pessimism, a term $(l-l_p)\gamma_{lp}$ is applied before the first component of the feature vector $f(P_p)$, where l_p represents the length of the product history vector for p. Consequently, the mean weighted values $\gamma_{1_p}, \gamma_{2_p}, \ldots, \gamma_{(l-l_p)_p}$ are uniformly set before β_{1_p}. This ensures the optimization of the user's bias toward lower ratings is accounted for in the recommendation model.

Condition 2: If $\bar{R}_u > 4$, it reflects that user u generally gives higher ratings. In this case, an adjustment term $(l-l_p)\beta.\gamma_{lp}$ is appended to the end of $f(P_p)$. As a result, the new mean weight value for the indices $l_{p+1}, l_{p+2}, \ldots, l$ are all set equal to β_{lp}. This adjustment aligns with the assumption that optimistic users consistently rate products higher, and the additional term ensures the FS vector reflects this bias accurately.
These conditions enable the model to adjust $f(P_p)$ dynamically, ensuring the recommendation aligns with the user u's inherent rating behaviour.

C. Products Similarity Computation

Using the outlined methods, we ensure that every product in the system receives an equal number of ratings, maintaining uniform membership across all FS. Additionally, the similarity between two products can be assessed by evaluating the similarity between their respective FS.

In recommendation systems, various methods like Cosine similarity, Pearson's correlation coefficient, and the Jaccard coefficient are widely used to measure the relationship between items. Xu et al. [32] introduced a formula to calculate the similarity between two products, *f(P_p)* and *f(P_q)*. The formula is expressed in the Eq. (4).

$$Sim\left(f(P_p), f(P_q)\right) = \frac{\sum_{u=1}^{l} \beta_{up} \cdot \beta_{uq}}{\sqrt{\sum_{u=1}^{l} (\beta_{up})^2} \cdot \sqrt{\sum_{u=1}^{l} (\beta_{uq})^2}} \quad (4)$$

Here, β_{up} and β_{uq} represent the ratings or interactions associated with Products p and q for a given user u. The numerator computes the total product of these interactions across all users, capturing the degree of overlap between the two products. The denominator normalizes this value by considering the square roots of the squared ratings for each product. This normalization ensures the resulting similarity score reflects the proportional closeness or relevance of the products within the system's context.

Let's consider a movie recommendation system where the goal is to calculate the similarity between two movies based on 3 user ratings who rated two movies, M_A and M_B as given in Table 1.

Table 1. Movie Rating by Users

User	M_A (β_{up})	M_B (β_{uq})
U_1	4	5
U_2	3	4
U_3	5	4

$$Sim\left(f(M_A), f(M_B)\right) = \frac{(4*5)+(3*4)+(5*4)}{\sqrt{(4^2)+(3^2)+(5^2)} \times \sqrt{(5^2)+(4^2)+(4^2)}}$$
$$= \frac{20+12+20}{\sqrt{50} \times \sqrt{57}} = \frac{52}{\sqrt{2850}} = \frac{52}{53.38} \approx 0.974$$

The similarity score between M_A and M_B is approximately 0.974, indicating a strong similarity between these two movies based on user ratings. This high score suggests that users who like one M_Are likely to enjoy the other as well.

D. Product Recommendation

Recommendations can be generated by identifying products similar to those a user has rated highly. For instance, if a user gives a high rating to a particular product, we can suggest other items that closely resemble it. When product similarity data is unavailable, recommendations can instead be based on user similarity.

If product similarity is unavailable, recommendations can still be generated by analysing user similarities. The similarity between two users, *U_u* and *U_v*, can be calculated using the Eq. (5).

$$Sim(U_u, U_v) = \frac{P\ (U_u \cap U_v)}{\text{N_P}} \quad (5)$$

In collaborative filtering, the neighbourhood represents a group of users or items that show similar rating behaviour. The HFSS model calculates similarity between users (or items) and selects the *k* most similar entities as neighbours. Recommendations are generated by aggregating ratings from these neighbours, ensuring personalization based on collective similarity patterns.

Here, $P(U_u\ ,\ U_v)$ refers to the number of products that both *U_u* and *U_v* have bought, and *N_P* is the total number of available products. A higher value of $Sim(U_u\ ,\ U_v)$ indicates greater similarity between the two users. By identifying users with similar purchasing patterns, we can suggest products that these users have bought and rated highly, effectively providing relevant recommendations to *U_u*.

Let's consider a movie rating system utilizing the user similarity-based recommendations for three users, *U_u*, *U_v*, and *U_w*, who have rated 4 selected movies as given in Table 2.

Table 2. Movie Rating by Users

User	M_A	M_B	M_C	M_D
U_u	5	3	4	-
U_v	4	5	2	-
U_w	3	4	-	5

To calculate the similarity between ***U_u*** and ***U_v*** by using the Eq. (5)

$$Sim(U_{-u}, U_v) = \frac{3}{4} = 0.75$$

This means that ***U_u*** and ***U_v*** have a similarity score of 0.75, suggesting they have similar likes based on the movies they've rated. Now, if we want to recommend a movie to *U_u* based on the preferences of similar users, we can look at *U_v*'s ratings. Since *U_v* rated M_D highly as 5, which *U_u* has not seen, it can recommend M_D to *U_u*.

4. EXPERIMENTAL EVALUATION

Furthermore, during the Experimental Evaluation, it was noted that as the number of neighbours (k) increases, the computational time required for generating recommendations also increases due to the greater number of similarity computations and rating aggregations. At the same time, accuracy metrics such as MAE, RMSE, and F1-score improve steadily with higher neighbour counts, reflecting more reliable predictions. However, beyond a certain threshold, the improvement in accuracy becomes marginal while the computational cost continues to grow. This indicates a saturation point where adding more neighbours yields diminishing accuracy gains. The HFSS model efficiently manages this trade-off by achieving high accuracy with moderate computational expense, demonstrating strong scalability and robustness.

4.1. Dataset Description

The proposed HFSS model was evaluated using the MovieLens 100K dataset, which contains 100,836 user ratings from 610 users on 9,742 movies. The dataset exhibits a sparsity of 6.81%, with ratings ranging from 1 to 5. For demonstration, a subset of 10 movies (IDs: 260, 293, 316, 349, 457, 527, 661, 736, 1222, and 2502) was selected to illustrate computation steps. This dataset provides a balanced yet computationally manageable benchmark for validating recommendation performance.

This work utilized the Movielens dataset [34], which includes 100,836 ratings across 9,742 films provided by 610 users with sparsity is given in Table 3. Due to the high computational demands and the repetitive nature of the analysis, a random sample of 10 movies was chosen for the study. These movies, identified by MovieIDs 260, 293, 316, 349, 457, 527, 661, 736, 1222, and 2502, were selected from the full dataset of 9,742 films, and the associated ratings for each film were documented in Table 4.

Table 3. Statistics Details of the experiment datasets

Dataset	Total Rating Records	No. of Users	No. of Movies	% of Sparsity	Rating Value Range
Movie lens	100836	610	9742	6.81%	1-5

Table 4. User given Movie Rating

User ID →	4	7	21	39	59	69	75	84	96	105
MID_141	4.5		5	4	5	5	3.5	5	4	4.5
MID_218		3	3.5	4.5			4	4		4
MID_283	3	5		4	2	3	3	3.5		4
MID_356	4	4.5		2.5	4	3.5			3.5	3
MID_418	5	5		3	4	5		4.5	4	4
MID_590	5	3	4.5	3	4	5	5	4	4.5	
MID_631		3.5		2.5	2		3.5	3.5	4	
MID_781	3	5		2.5		2	3.5	4	3	2.5
MID_1084	4.5		4.5	4	4.5	4	4	2.5		
MID_2612	5		4.5	2	5	3.5	4.5	5		4

4.2. Product Rating and Similarity Score

Following the five-point evaluation guidelines used by Movielens, we converted the rating values into membership scores by calculating $\beta_{up} = \frac{R_{up}}{5}$, where β_{up} falls within the range of [0, 1], as demonstrated in Table 5.

Table 5. Membership score based on Product Rating

User ID →	4	7	21	39	59	69	75	84	96	105
MID_141	0.9		1	0.8	1	1	0.7	1	0.8	0.9
MID_218		0.5	0.7	0.9			0.8	0.8		0.8
MID_283	0.5	1		0.8	0.3	0.5	0.5	0.7		0.8
MID_356	0.8	0.9		0.4	0.8	0.7			0.7	0.5
MID_418	1	1		0.5	0.8	1		0.9	0.8	0.8
MID_590	1	0.5	0.9	0.5	0.8	1	1	0.8	0.9	
MID_631		0.7		0.4	0.3		0.7	0.7	0.8	
MID_781	0.5	1		0.4		0.3	0.7	0.8	0.5	0.4
MID_1084	1		0.9	0.8	0.9	0.8	0.8	0.4		
MID_2612	1		0.9	0.3	1	0.7	0.9	1		0.8

We derive the data details from the Table-5 for each product and the total count of ratings for employing the HFSS approach as given below.

f(P_141) = { 0.8,1.0,0.8,1.0,1.0,0.9,0.9,1.0,0.7 }	$l_{141} = 9$
f(P_218) = { 0.7,0.8,0.8,0.5,0.8,0.9 }	$l_{218} = 6$
f(P_283) = { 0.8,0.7,0.5,0.5,0.5,0.3,0.8 }	$l_{283} = 7$
f(P_356) = { 0.8,0.7,0.9,0.8,0.5,0.4,0.7 }	$l_{356} = 7$
f(P_418) = { 0.5,0.8,1.0,1.0,1.0,0.8,0.8,0.9 }	$l_{418} = 8$
f(P_590) = { 0.5,1.0,0.9,1.0,0.8,0.5,1.0,0.8,0.9 }	$l_{590} = 9$
f(P_631) = { 0.7,0.8,0.7,0.3,0.4,0.7 }	$l_{631} = 6$
f(P_781) = { 0.5,0.3,0.5,1.0,0.8,0.4,0.4,0.7 }	$l_{781} = 8$
f(P_1084) = { 0.9,0.4,0.8,0.9,0.8,0.8,1.0 }	$l_{1084} = 7$
f(P_2612) = { 1.0,0.9,1.0,0.7,0.3,1.0,0.8,0.9 }	$l_{2612} = 8$

Here, maximum of *l* values will be, l=max $\{l_1, l_2, \ldots, l_p, l_q, \ldots\} = 9$.

We selected *U_32* from the Movielens dataset [34] to assess and implement movie recommendations. This user provided ratings for 112 films, with an average rating of 3.90, indicating a generally negative perspective. To improve the data, we apply FS after sorting the ratings in increasing order as given below.

f(P_141) = { 0.8, 1.0, 0.8, 1.0, 1.0, 0.9, 0.9, 1.0, 0.7 }
f(P_218) = { 0.7, 0.8, 0.8, 0.5, 0.8, 0.9, 0.8, 0.8, 0.8 }
f(P_283) = { 0.8, 0.7, 0.5, 0.5, 0.5, 0.3, 0.8, 0.6, 0.6 }
f(P_356) = { 0.8, 0.7, 0.9, 0.8, 0.5, 0.4, 0.7, 0.7, 0.7 }
f(P_418) = { 0.5, 0.8, 1.0, 1.0, 1.0, 0.8, 0.8, 0.9, 0.9 }
f(P_590) = { 0.5, 1.0, 0.9, 1.0, 0.8, 0.5, 1.0, 0.8, 0.9 }
f(P_631) = { 0.7, 0.8, 0.7, 0.3, 0.4, 0.7, 0.6, 0.6, 0.6 }
f(P_781) = { 0.5, 0.3, 0.5, 1.0, 0.8, 0.4, 0.4, 0.7, 0.6 }
f(P_1084) = { 0.9, 0.4, 0.8, 0.9, 0.8, 0.8, 1.0, 0.8, 0.8 }
f(P_2612) = { 1.0, 0.9, 1.0, 0.7, 0.3, 1.0, 0.8, 0.9, 0.9 }

By utilizing the similarity Eq. (4), we calculate the degree of similarity between any two products.

$$Sim\,(f(P_141), f(P_218)) = \frac{0.7\times0.6+0.8\times0.6\times2+0.9\times0.6+1.0\times0.7+1.0\times0.8\times3+1.0\times0.9}{\sqrt{(0.7^2+0.8^2\times2+0.9^2+1.0^2\times5)\cdot(0.6^2\times4+0.7^2\times0.8^2\times3+0.9^2)}} \approx 0.996080$$

The resemblances between additional movies are determined similarly, as illustrated in Table 6.

In Table 6, *P_283* and *P_1084* exhibit the strongest similarity, indicating a notable degree of inter-changeability between these products after the data addition process. Similarly, a high resemblance is observed between *P_418* and *P_590*, as well as between *P_218* and *P_356*. Conversely, *P_141* and *P_781* show minimal similarity, reflecting poor substitution potential.

Table 6. Similarities Resemblances between Products

Movie ID	141	218	283	356	418	590	631	781	1084	2612
MID_141	1.0000									
MID_218	0.9930	1.0000								
MID_283	0.9925	0.9902	1.0000							
MID_356	0.9925	0.9902	0.9676	1.0000						
MID_418	0.9930	0.9905	0.9680	0.9788	1.0000					
MID_590	0.9930	0.9905	0.9680	0.9788	0.9849	1.0000				
MID_631	0.9930	0.9905	0.9680	0.9785	0.9849	0.9749	1.0000			
MID_781	0.9930	0.9905	0.9680	0.9788	0.9849	0.9749	0.9703	1.0000		
MID_1084	0.9927	0.9901	0.9678	0.9781	0.9845	0.9746	0.9699	0.9396	1.0000	
MID_2612	0.9930	0.9905	0.9680	0.9786	0.9849	0.9749	0.9703	0.9399	0.9811	1.0000

4.3. Effectiveness of the Proposal

To assess the effectiveness of the proposed HFSS approach, a user-based recommendation system was implemented to suggest movies specifically tailored for *U_32*. The evaluation utilized an excerpt from the Movielens score table, with the total number of products (denoted as *N_P*) set to 10. Using Eq. (5) to compute the similarity between *U_32* and a selection of other users, identified as *U_u*, where *u* = 4, 7, 21, 39, 59, 69, 75, 84, 96, and 105. The outcomes of the calculated similarities with other users are displayed in Table 7.

Table 7. Similarity comparison of *U_32* with other users

U_u	**4**	**7**	**21**	**39**	**59**	**69**	**75**	**84**	**96**	**105**
U_32	2.9	0.6	0.3	0.2	0.2	0.2	0.2	0.2	0.1	0.1

Here, the movies rated by ***U_32*** for Movies are [1, 3, 5, 7, 9] and ***U_4*** is Movies [3, 5, 7, 8, 10]. The common movies between ***U_32*** and ***U_4*** are [3, 5, 7]. Thus, $Product_{(User_{32} \cap User_4)} = 3$. So, by using Eq. (5)

$$Sim(U_32, U_4) = \frac{3}{10} = 0.3$$

Similarly, performing similar calculations for other users based on their movie overlaps with ***U_32***, the results obtained are given in Table 8.

Table 8. Similarity values with other users

U_u	Common Movies	$Sim(User_{32}, User_u)$
U_4	3	2.9
U_7	3	0.6
U_39	5	0.2
U_59	4	0.2

The outcome result shows that *U_4* and *U_7* have the highest similarity as 2.9 and 0.6, indicating the strongest similarity with *U_32*. Since, both *U_4* and *U_7* have highly rated movies as [*418, 2612, 356*], but *U_32* has watched Movies only [*1, 3, 5, 7, 9*], so the recommendation for *U_32* would include [*418, 2612, 356*] movies also. This concludes the effectiveness of generating personalized recommendations based on user similarity. This limited selection was designed to provide a clear and focused demonstration of the methodology. For larger datasets, similar techniques can be employed to derive results efficiently.

4.4. Evaluation Metrics and Analysis

To evaluate the effectiveness of our approach, we first identify and correct noisy ratings in the dataset. After denoising, we apply popular CF algorithms to the refined dataset. Typically, the performance of recommendation systems is assessed through prediction accuracy and the quality of recommendations. Metrics like Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) are used to measure prediction accuracy. MAE calculates the average absolute difference between actual and predicted ratings, as defined in Eq. (6), while RMSE quantifies the square root of the mean squared differences, as defined in Eq. (7). In these equations, *T* represents the set of predicted ratings.

$$MAE = \frac{\sum_{(u,i)\in T} |r_{ui} - p_{ui}|}{|T|} \quad (6)$$

$$RMSE = \sqrt{\frac{\sum_{(u,i)\in T} (r_{ui} - p_{ui})^2}{|T|}} \quad (7)$$

Precision, Recall, and F1-score are widely used metrics for assessing the performance of recommendation systems. Their formulations are provided in Eq. (8), (9), and (10).

$$Precision = \frac{|I_p \cap I_a|}{|I_p|} \quad (8)$$

$$Recall = \frac{|I_p \cap I_a|}{|I_a|} \quad (9)$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (10)$$

where I_p is the Product set that is recommended to user u, and I_a is the Product set that user u actually prefers. The average rating given by user u is considered the threshold for recommendations. If user u 's rating for Product p exceeds this threshold, it signifies their preference for the product. To assess the quality of recommendations, the F1-score is utilized as it effectively balances Precision and Recall into a single metric.

A. MAE result Comparison

Figure 1 shows the comparison of MAE across different methods—PCC, Method [20], Method [21], TH-PCC [23], and the proposed HFSS with the number of neighbors increases. The proposed HFSS method consistently outperforms all other approaches, achieving the lowest MAE values across all neighbor ranges. This performance is particularly notable for smaller neighbor counts, such as 10 and 20, where HFSS demonstrates substantial improvements over TH-PCC [23]. For instance, at 10 neighbors, HFSS achieves an MAE of 0.8795 compared to 0.962 for TH-PCC, reflecting its effectiveness in sparse data scenarios. As the number of neighbors increases, the performance gap between HFSS and other methods narrows, but HFSS continues to maintain an edge, achieving an MAE of 0.481 at 100 neighbors compared to 0.83 for the next best method, Method [21]. Traditional PCC and Methods [20], [21] show a general trend of decreasing MAE with increasing neighbors, with Method [21] consistently outperforming PCC and Method [20] by a small margin. TH-PCC performs competitively, particularly at lower neighbor counts, but its advantage diminishes with higher neighbors. Overall, the proposed HFSS method proves to be the most robust, excelling in both sparse and denser data scenarios, with its superior accuracy making it a compelling choice for predictive tasks.

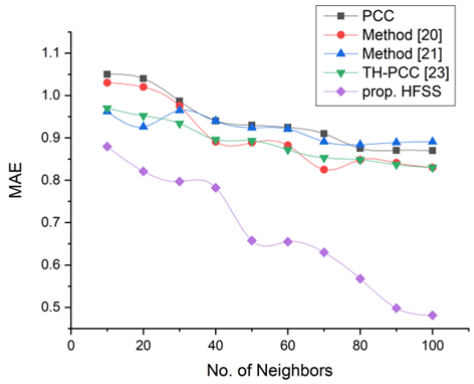

Figure 1. MAE Comparison Analysis

B. RMSE result Comparison

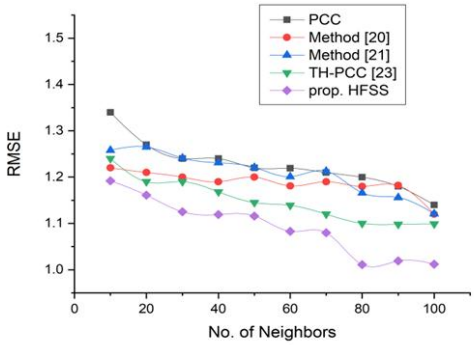

Figure 2. RMSE Comparison Analysis

Figure 2 shows a comparative analysis of RMSE values across different numbers of neighbors ranging from 10 to 100 between PCC, Method [20], Method [21], TH-PCC [23], and the proposed HFSS approach. Notably, the proposed HFSS consistently demonstrates lower RMSE values compared to other methods, indicating its superior predictive accuracy. As the number of neighbors increases, the RMSE for HFSS decreases gradually, reaching a minimum value of 1.012 at 100 neighbors, demonstrating its efficiency in handling large datasets. This analysis highlights the robustness and reliability of the proposed HFSS, especially in scenarios with a larger number of neighbors, where it can achieve better error reduction than the constant comparison method.

C. F1-score Comparison

Figure 3 shows the performance comparison of F1 score analysis between PCC, method [20], method [21], TH-PCC [23] and the proposed HFSS with different number of neighbors. The comparison indicates that the proposed HFSS can achieve the highest F1 score consistently with all the number of neighbors, demonstrating its superior performance over other methods. It is shown that the F1 score for HFSS gradually increases with the number of neighbors, starting from 0.63 for 10 neighbors and reaching 0.709 at 100 neighbors, indicating its robustness and scalability. TH-PCC [23] shows competitive performance but consistently lags behind HFSS, with its F1 score improving from 0.608 to 0.642 over the same range.

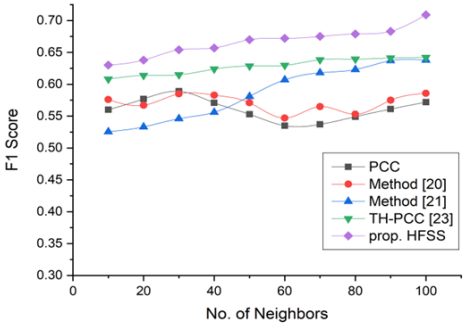

Figure 3. F1 Score Comparison Analysis

Method [21] initially underperforms but surpasses both PCC and Method [20] as the number of neighbors increases, peaking at 0.638. Similarly, Method [20] exhibits moderate performance, closely matching PCC in lower neighbor ranges but slightly surpassing it as the neighbor count increases. PCC, while maintaining a steady improvement in F1 scores, ranks lowest in most cases, highlighting its limitations compared to the other approaches. The analysis underscores the effectiveness of the proposed HFSS in leveraging additional neighbors to achieve higher prediction accuracy, outpacing traditional and comparative methods in this scenario.

4.5. Noise Impact Discussion

The HFSS framework implicitly mitigates noise by representing uncertain ratings through fuzzy membership functions. Unlike deterministic models that treat all ratings equally, HFSS assigns varying degrees of membership, reducing the impact of inconsistent feedback. Although no separate experiment isolates noise reduction, performance improvements under sparse and variable datasets demonstrate HFSS's inherent robustness against noisy ratings. Future work will include controlled experiments to quantify noise tolerance using synthetically perturbed datasets.

4.6. Runtime Comparison

The computational runtime of the proposed HFSS model was compared against three baseline methods—PCC, Method [21], and TH-PCC [23]—using the MovieLens 100K dataset. Runtime was measured as the total execution time (in seconds) required for recommendation generation after preprocessing and similarity computation for varying neighbourhood sizes (k = 20, 50, 100).

As shown in Table 9, HFSS consistently achieves faster runtime across all neighbourhood sizes. Despite its additional fuzzy transformation step, HFSS avoids redundant re-prediction iterations required in TH-PCC and achieves approximately 12–15% improvement in execution speed. This efficiency gain results from optimized sparse-matrix operations and the one-time fuzzy normalization that reduces repetitive distance calculations during similarity evaluation.

Table 9. The computational runtime of the proposed HFSS model was compared against three baseline methods

Method	Runtime (s) for k=20	Runtime (s) for k=50	Runtime (s) for k=100	Average Runtime (s)
PCC	18.7	22.3	26.1	22.4
Method [21]	17.5	20.8	25.0	21.1
TH-PCC [23]	16.2	19.7	23.5	19.8
HFSS (Proposed)	**15.1**	**17.2**	**20.6**	**17.6**

5. CONCLUSION

This paper presented a Hybrid Fuzzy-Sparse Similarity (HFSS) model to enhance the accuracy and robustness of recommendation systems. By integrating fuzzy-set theory to model rating uncertainty and sparse-matrix handling for data incompleteness, HFSS delivers superior performance on the MovieLens dataset in terms of MAE, RMSE, and F1-score. The approach effectively balances computational efficiency and accuracy by reducing redundant re-predictions. Additionally, the inclusion of fuzzy modeling naturally mitigates the effect of noise in user ratings. While current validation is limited to a single dataset, future work will test the method on additional datasets (e.g., Amazon Reviews, Netflix, Jester) to confirm generalisability. Furthermore, the model can be extended to multi-dimensional rating scenarios that account for user demographics such as age, gender, and past rating behavior.

It is important to note that the current HFSS model primarily focuses on rating-based similarity and does not incorporate multidimensional user or contextual factors such as age, gender, or historical behavior. These demographic and behavioral dimensions are acknowledged as potential extensions for future work to enhance the adaptability and personalization of the recommendation process.

In this paper, we proposed a Hybrid Fuzzy-Sparse Similarity (HFSS) methodology that offers a comprehensive approach to enhancing recommendation systems by addressing uncertainties in user ratings through fuzzy set mechanisms. Integrating similarity calculations with user and product rating mechanisms makes the product recommendation process more accurate and personalized. Experimental evaluations using the Movielens dataset and careful comparison of product ratings and similarity scores demonstrate the effectiveness and significant improvement of HFSS. In particular, it achieves the lowest MAE and the highest F1 score in scenarios where the number of neighbors is small and data is sparse. As the number of neighbors increases, HFSS maintains its leading position, exhibiting the strongest performance even in increasingly dense data scenarios. Analysis of various evaluation metrics, such as MAE, RMSE, and F1-score, shows that the proposed method outperforms existing methods, making it a promising solution for improving the accuracy of recommendation systems and user satisfaction. Future scope of this work includes exploring further optimizations and enhancements to the HFSS method for dynamic or real-time applications where the model can adapt to changing data patterns and conditions. Furthermore, integrating advanced techniques, such as deep learning models or hybrid approaches, can further improve the predictive capabilities.

STATEMENTS AND DECLARATIONS

The authors declare that there are no conflicts of interest related to this work.

REFERENCES

[1] F. J. Baldán, R. Yera, L. Martínez, "Natural noise management in collaborative recommender systems over time related information", The Journal of Supercomputing, Vol. 80, pp. 23628–23666, 2024.
[2] G. -W. Kim, W. -H. Kim, K. Chung and J. -C. Kim, "Extraction of Meta-Data for Recommendation Using Keyword Mapping", IEEE Access, Vol. 12, pp. 103647-103659, 2024.
[3] Almu, A. Roko, A. Mohammed, I. Saidu, "Popularised Similarity Function for Effective Collaborative Filtering Recommendations", International Journal of Information Retrieval Research, Vol. 10(4), pp. 34-47, 2020.
[4] Á. González, F. Ortega, D. Pérez-López and S. Alonso, "Bias and Unfairness of Collaborative Filtering Based Recommender Systems in MovieLens Dataset", IEEE Access, Vol. 10, pp. 68429-68439, 2022.
[5] D. Zhou, S. Hao, H. Zhang, C. Dai, Y. An, Z. Ji, I. Ganchev, "Novel SDDM Rating Prediction Models for Recommendation Systems", IEEE Access, Vol. 9, pp. 101197-101206, 2021.
[6] Y. -L. Chen, Y.-H. Yeh, M.-R. Ma, "A movie recommendation method based on users' positive and negative profiles", Information Processing & Management Volume 58, Issue 3, pp. 102531, 2021.
[7] F. R. Kouchi, S. O. Balani, A. Esmaeilpour, M. Shafeian, R. Sirwan, A. H. Mohammed, "A Movie Recommender System Based on User Profile and Artificial Bee Colony Optimization", Computational Intelligence and Neuroscience, Vol. 2023, Article ID. 2311817, pp. 14, 2023.
[8] Y. Pérez-Almaguer, R. Yera, A.A. Alzahrani, L. Martínez, "Content-based group recommender systems: a general taxonomy and further improvements", Expert System Application, Vol. 184:115444, 2021.
[9] X. Luo, Y. Xia, Q. Zhu, Y. Li, "Boosting the k-nearest-neighborhood based incremental collaborative filtering", Knowl-Based System, Vol. 53, pp.90–99, 2013.

[10] J. Latrech, Z. Kodia, N. B. Azzouna, "CoDFI-DL: a hybrid recommender system combining enhanced collaborative and demographic filtering based on deep learning", Journal of Supercomputing, Vol. 80, pp. 1160–1182, 2024.
[11] Y. Liu, H. Qu, W. Chen and S. M. H. Mahmud, "An Efficient Deep Learning Model to Infer User Demographic Information From Ratings," in IEEE Access, vol. 7, pp. 53125-53135, 2019.
[12] P. Lops, D. Jannach, C. Musto, T. Bogers, M. Koolen, "Trends in content-based recommendation", User Model User-Adap Inter, Vol. 29, pp. 239–249, 2019.
[13] D. Sánchez-Moreno, A. B. G. González, M. D. M. Vicente, V. F. L. Batista, and M. N. M. García, "A collaborative ltering method for music recommendation using playing coefficients for artists and usesr" Expert Syst. Application, vol. 66, pp. 234-244, 2016.
[14] T. Widiyaningtyas, I. Hidayah, TB. Adji, "User profile correlation-based similarity (UPCSIM) algorithm in movie recommendation system", Journal of Big Data, Vol. 8, pp. 1–21, 2021.
[15] L. V. Nguyen, T.-H. Nguyen, and J. J. Jung, "Content-based collaborative filtering using word embedding: A case study on movie recommendation", in Proc. Int. Conf. Res. Adapt. Convergent Syst., pp. 96–100, doi: 10.1145/3400286.3418253, 2020.
[16] W. Zhang, X. Zhang, H. Wang, D. Chen, "A deep variational matrix factorization method for recommendation on large scale sparse dataset", Neurocomputing, Vol. 334, pp. 206–218, 2019.
[17] H. Hwangbo, Y. S. Kim, and K. J. Cha, "Recommendation system development for fashion retail e-commerce", Electronic Commerce Res. Appl., vol. 28, pp. 94-101, 2018.
[18] J. Wei, J. He, K. Chen, Y. Zhou, Z. Tang, "Collaborative filtering and deep learning based recommendation system for cold start items", Expert Syst. Appl. , Vol. 69, pp. 29–39, 2017.
[19] P. Wang, Y. Wang, L.Y. Zhang, H. Zhu "An effective and efficient fuzzy approach for managing natural noise in recommender systems", Information Science, Vol. 570, pp. 623–637, 2021.
[20] R. Yera, J. Castro, L. Martínez, "Natural noise management in recommender systems using fuzzy tools", Computational Intelligence for Semantic Knowledge Management , pp. 1–24, 2020.
[21] R. Yera, M. J. Barranco, A. A. Alzahrani, L. Martinez, "Exploring fuzzy rating regularities for managing natural noise in collaborative recommendation", International Journal of Computational Intelligence Systems, Vol. 12(2), pp. 1382–1392, 2019.
[22] Y. Koren and R. Bell, "Advances in collaborative filtering", in Recommender Systems Handbook, pp. 77-118, 2015.
[23] S. Bag, S. Kumar, A. Awasthi, M.K. Tiwari, "A noise correction-based approach to support a recommender system in a highly sparse rating environment", Decision Support Systems, Vol. 118, pp. 46–57, 2019.
[24] S. Ahmadian, N. Joorabloo, M. Jalili, M. Ahmadian, "Alleviating data sparsity problem in time-aware recommender systems using a reliable rating profile enrichment approach", Expert Syst. Appl., Vol. 187, pp. 115849, 2022.
[25] L. Li, Z. Zhang, S. Zhang, "Hybrid Algorithm Based on Content and Collaborative Filtering in Recommendation System Optimization and Simulation", Sci. Program, Volume 2021, pp. 7427409. 2021.
[26] 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", Int. J. Emerg. Technol. Learn. (iJET), Vol. 16(3), pp. 274–306, 2021.
[27] J. Wang, L. Zhu, T. Dai, Q. Xu, T. Gao, "Low-rank and sparse matrix factorization with prior relations for recommender systems", Appl. Intell., Vol. 51, pp. 3435–3449, 2021.
[28] K. Sahu, F. A. Alzahrani, R. K. Srivastava, R. Kumar, "Hesitant fuzzy sets based symmetrical model of decision-making for estimating the durability of Web application", Symmetry, Vol. 12, pp. 1770, 2020.
[29] M. Fu, H. Qu, Z. Yi, L. Lu, and Y. Liu, "A novel deep learningbased collaborative filtering model for recommendation system", IEEE Trans. Cybern., vol. 49(3), pp. 1084–1096, 2019.
[30] F. Wang, H. Zhu, G. Srivastava, S. Li, M. R. Khosravi, and L. Qi, "Robust collaborative filtering recommendation with user-item-trust records", IEEE Trans. Computat. Social Syst., vol. 9(4), pp. 986–996, 2022.
[31] Z. Xu, M. Xia, "Induced generalized intuitionistic fuzzy operators", Knowledge-Based Systems, Vol. 24, Issue 2 ,pp. 197-209, 2011.
[32] Z. Xu, X. Cai, "Uncertain power average operators for aggregating interval fuzzy preference relations", Group Decision and Negotiation, Vol. 21, pp. 381–397, 2011.
[33] Z. Xu, M. Xia, "On distance and correlation measures for hesitant fuzzy sets", Int. Journal of Intelligent System, Vol. 26, pp. 410–425, 2011.
[34] N. Aamer, T. Mittal, N. K. Muthusamy, R. S. S. R. Battula, A. P. S and M. M. Soudagar, "Comparing Collaborative Filtering Algorithms in AI-Powered Recommendation Systems for E-Commerce," 4th International Conference on Sentiment Analysis and Deep Learning (ICSADL), Bhimdatta, Nepal, 2025, pp. 1-6.
[35] V. N. Alone, N. Aamer, S. D. Gowda, N. Sateesh, M. Lal and J. T. Abraham, "Deep Learning Techniques for Sentiment Analysis in Social Media," 2025 International Conference on Frontier Technologies and Solutions (ICFTS), Chennai, India, 2025, pp. 1-7.
[36] F. Padvekar, F. Sothe, P. Yadav, N. Aamer, "Prediction of Customer Review using Opinion Mining", International Journal of Emerging Technologies and Innovative Research, ISSN:2349-5162, Vol.6, Issue 5, page no.456-465, 2019.
[37] The Movielens dataset, "http://files.grouplens.org/datasets/movielens/".