Towards Improved Movie Recommendations via Handling of Input Uncertainty

Prodipta S. Amartya*, Tahmid Imtiaz*, Shaily Kabir†, Christian Wagner†
*Department of Computer Science and Engineering, University of Dhaka, Bangladesh
†Lab for Uncertainty in Data and Decision Making (LUCID), School of Computer Science, University of Nottingham, UK
Email:{prodiptasen-2018325324,tahmidimtiaz-2018125317}@cs.du.ac.bd, {shaily.kabir5,christian.wagner}@nottingham.ac.uk

***Abstract*—Movie recommendation systems have greatly advanced through AI and machine learning (ML), but challenges with recommendation accuracy still persist due to uncertain and noisy data. In this regard, singleton fuzzy logic systems (SFLSs) with crisp inputs and outputs offer simple and cost-effective solutions for ambiguous user preferences. However, SFLSs exhibit limitations in handling noisy and uncertain inputs—such as caused by vague or incomplete/wrong text descriptions of movies and ratings, risking misleading recommendations. In this context, non-singleton FLSs (NSFLSs) that use fuzzy sets (FSs) to capture input uncertainties have the potential to enhance robustness and accuracy of the recommendation systems. This paper investigates an NSFLS-based movie recommendation system and presents a comparative analysis of its performance against an SFLS-based system. Experiments conducted on a real-world movie ratings data set with varying degrees of noise reveal NSFLSs as a useful approach to enhance the accuracy of movie recommendations with noisy data.**

***Index Terms*—singleton fuzzy logic system, non-singleton, vagueness, uncertainty**

I. Introduction

Movie recommendation systems aim to offer personalized movie suggestions to users considering their preferences and movie features, making it easier to navigate through the vast collection of movies [1]. Current trends in movie recommendations reflect advances in AI, ML, and big data analytics, but challenges still remain. One of the primary challenges is to provide reliable recommendations for uncertain and noisy data sets [2], [3]. In this regard, the fuzzy logic system (FLS) is often used to design a reliable movie recommendation framework for ambiguous user ratings. The authors in [4] developed a simple and cost-effective type-1 FLS-based movie recommendation system with the MovieLens data sets [5]. In addition, the authors in [6] also presented a movie recommendation system considering interval type-2 FLS that efficiently manages the uncertainty of the data and improves the accuracy of calculating the expected rating of a movie for a target user. All of these works demonstrate how well singleton FLSs (SFLSs) with crisp inputs and outputs handle user preferences and generate solutions. However, these recommendation systems are developed considering only noise-free movie ratings. SFLSs are not designed to systematically capture and handle input uncertainty [7], highlighting potential room for improvement with alternative systems.

In real-world applications, noisy data is prevalent and can easily be generated from geographical fluctuations, such as regional restrictions, vague data labeling of participants, or incorrect text representation [8]. With noisy data, the performance of SFLSs tends to degrade [9], [10]. To illustrate this, consider an FLS-based movie recommendation system in Fig. 1 (adapted from [10]) where noise causes the input rating x' to change to x''. If the recommendation system is based on SFLS, it does not have the means to capture this uncertainty and is constrained to proceed with the numeric rating x'', as a singleton input, misclassifying a 'High' rating as 'Low' rating with a degree of membership $\mu_s(x'')$; it eventually selects inappropriate fuzzy rules for fuzzy inference and yields an incorrect outcome. From this example, it is clear that we need an alternative approach to better handle noise and enhance the accuracy of FLSs.

A non-singleton FLS (NSFLS), an extension of SFLS, is designed to capture the uncertainties associated with the input of an FLS using fuzzy sets (FSs) [9], [11], [12]. Figure 1 shows how an NSFLS-based movie recommendation system can handle input noise. In this case, the noisy input x'' is modeled using a FS (here, we use a Gaussian FS) and using the standard input firing method (details in Section II-B), x'' is evaluated with respect to the 'High' membership function (MF) at 'B' with a membership degree of $\mu_{ns_2}(x'')$ along with the 'Low' MF at 'D' with $\mu_{ns_1}(x'')$. The NSFLS triggers the firing strengths for both 'Low' and 'High' MFs and mitigates the effect of noise. Hence, NSFLSs have the ability to enhance the performance of FLSs by reducing the impact of noise and inconsistency in the input data.

In this context, this paper investigates the application of NSFLSs to develop movie recommendation systems for noisy

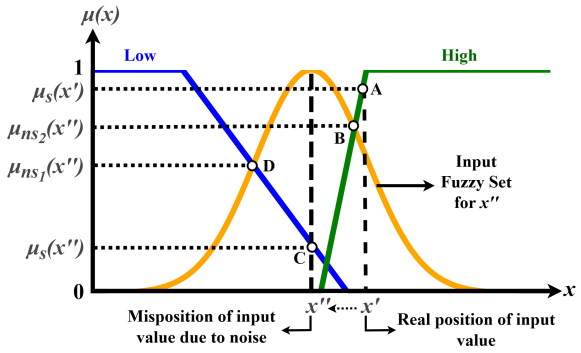


Fig. 1: Handling of input uncertainty using an NSFLS

ratings. It presents the entire process of managing input data in noisy situations using various fuzzy inference systems (FIS). It also demonstrates a comparative analysis of the performance of the proposed NSFLS-based movie recommendation systems with that of SFLS-based recommendation system with respect to a real-world movie ratings data set having various levels of noise, discovering NSFLSs as a useful method to improve the move recommendation accuracy.

The rest of the paper is organized as follows. Section II presents a brief discussion on various methodologies used for recommendation systems along with the benefits of including FLSs to recommend movies and a synopsis of SFLS and NSFLS. Section III introduces the working procedure of a movie recommendation system with respect to SFLS and NSFLS, and Section IV demonstrates the performance analysis of SFLS and NSFLS-based movie recommendation systems. Lastly, Section V concludes the paper along with future work.

II. Background

This section briefly discusses the methodologies behind recommendation systems, highlighting the benefit of incorporating FLSs, particularly NSFLSs,

A. Methodologies Used for Recommendation Systems

In recent years, the increasing challenge of information overload has led to the emergence of recommendation systems that enable users to efficiently navigate and filter large volumes of data [13]. These systems are particularly designed to prioritize information tailored to individual user preferences, focusing on content that is most relevant or likely to be of interest. By doing so, they alleviate the burden of information overload, reducing the time and effort that users must spend searching through excessive data.

Following widespread adoption of recommendation systems, content-based filtering (CBF) has emerged as a method that recommends new items to users based on similar items they chose earlier, i.e. their usage profiles [14]. In other words, other user's preferences have no impact on the recommendations to a target user. This approach demonstrates adaptability by quickly adjusting recommendations to reflect changes in an individual user's profile, maintaining relevance over time. However, CBF has a notable drawback; it requires a comprehensive and explicit representation of the characteristics of the objects in the user profile, which can pose significant challenges in practical applications [15]. In addition to CBF, collaborative filtering (CF), another approach, recommends items to users considering the interactions of other users who have similar preferences and behaviour [16]. Traditional CF also faces limitations in managing scalability and maintaining accuracy [17]. In particular, these limitations are evident when dealing with sparse or incomplete data sets, making it more difficult for the recommendation system to provide effective and reliable recommendations [18].

To address the challenges of CBF and CF, approaches such as matrix factorization (MF) [19] and deep learning (DL) [20] were introduced. MF algorithms decompose user-item interaction matrices into latent factors, capturing patterns to predict missing ratings and generate personalized recommendations, improving the accuracy of recommendations [19]. On the other hand, DL algorithms—which use neural networks—are adept at managing unstructured data, sequential behaviors, and diverse tasks and provide users with personalized and context-aware recommendations by processing vast data sets capturing user behavior [20]. Nevertheless, these approaches often involve slow and resource-intensive training procedures and have trouble scaling effectively [21]. In addition, these models associated with the implementation of ML show difficulties in explaining the overall system in a more interpretable way close to human understanding [22]. In the case of ML models, more interpretability often leads to the sacrifice of the accuracy of the outcome [23]. In addition, users often prefer to express their preferences or ratings linguistically, rather than using numerical scores [24]. In this case, since ML models rely heavily on precise user ratings, their performance deteriorates with such fuzzy preference [25].

Effective handling of user richer-than-numeric (e.g. interval-valued) feedback is crucial to increasing recommendation accuracy because it not only enhances the assessment of user similarity and neighbor selection, but also reduces prediction errors [13]. The integration of FLS into recommendation systems has shown significant advantages [26]. By capturing nuanced user preferences through FSs, FLSs enable accurate user modeling beyond simple ratings. Furthermore, FLSs have better functionality and tools to interpret their sophisticated structure used in a recommendation system [22]. Compared to conventional techniques, fuzzy distance metrics improve similarity computation by better handling of diverse and imprecise user features [27]. Incorporating FLSs into a recommendation system has the potential to produce better recommendation accuracy with reduced recommendation error and stable coverage, thus ensuring a balanced and practical recommendation system [18].

In case of movie recommendation, SFLS-based recommendation models have already achieved most of the functionalities mentioned above [4], [6], [28]. However, a noisy or uncertain input can significantly deteriorate their performance [29], thus clearly presenting a need to establish a movie recommendation system that can improve the accuracy and reliability of movie recommendations for uncertain and noisy data. NSFLSs can address input uncertainties through non-singleton fuzzification by converting crisp inputs into fuzzy input sets, unlike singleton fuzzification (with crisp values) of SFLSs [11], [30], showing a promising approach for developing movie recommendation systems with noisy data.

B. Singleton and Non-singleton FLSs

We present a brief overview of SFLS and NSFLS along with key NSFLS methods for determining the firing strength under input uncertainties: the standard composition- and centroid-based approaches. It is noted that an SFLS and an NSFLS

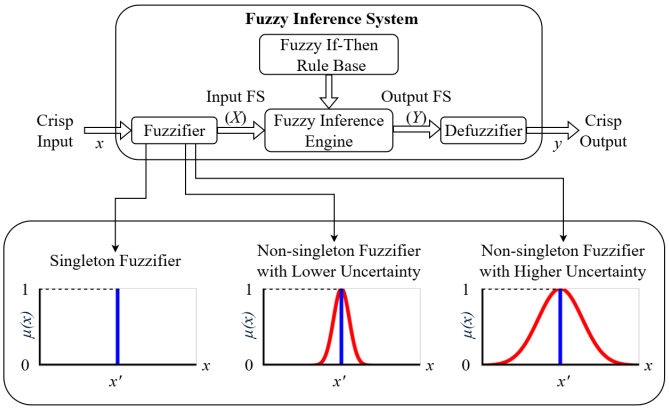


Fig. 2: Overview of FLS structure with singleton and non-singleton Gaussian input FSs

share identical general structure of FIS [31], as shown in Fig. 2, adapted from [11], [31].

1) Singleton FLS (SFLS) : In the SFLS, the fuzzifier maps crisp, i.e. numeric inputs into singleton fuzzy sets [31], defined in (1),

$$\mu_X(x_i) = \begin{cases} 1, & x_i = x_i' \\ 0, & x_i \neq x_i'. \end{cases} \tag{1}$$

where x_i' is the crisp input and x_i are the neighbouring values. Note that the singleton fuzzification is calculating $\mu_X(x_i)$ for a given value of x_i to determine the firing strength.

2) Non-Singleton FLS (NSFLS) : In a NSFLS, either a fuzzy-set valued input is used directly, or non-singleton fuzzification is used to transform a crisp input into an input FS [31]. Note that the MF used for the input FS is application-dependent, commonly a convex, normal FS [9]. Figure 2, shows an example of applying the Gaussian distribution as the input FS. Equation (2) defines a non-singleton Gaussian fuzzifier [9].

$$\mu_X(x_i) = \exp\left(-\frac{(x_i - x_i')^2}{2\sigma_F^2}\right), \tag{2}$$

where, x_i' is the crisp input and the mean value of the input FS, $\mu_X(x_i)$ is the degree of membership, and σ_F is the spread of input FS whereas x_i are the neighboring values of the mean. A gaussian with a larger standard deviation is generally associated with a higher degree of uncertainty in the input [11]. There are different approaches to define the composition between input and antecedent FSs within non-singleton fuzzy inference [9], [29]. In this paper, we consider the standard composition- and centroid-based non-singleton fuzzifiers, discussed below.

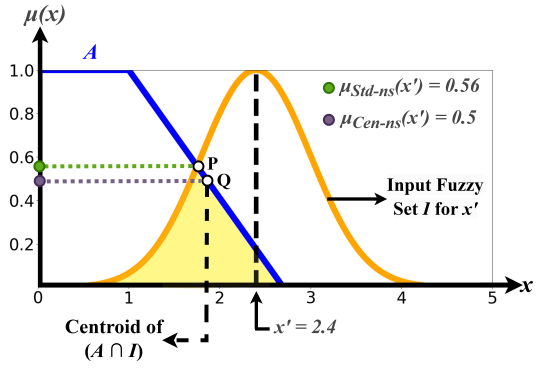


Fig. 3: Standard composition- and centroid-based firing strength of the NSFLSs

- *NSFLSs with standard composition-based inference:* In the standard NSFLS composition, the firing strength is defined as the maximum degree of membership of the intersection between the input FS, I and the antecedent FS, A [29]. Consider the example in Fig. 3 where I for x' = 2.4 intersects A at 'P' with a membership degree of $\mu_{Std-ns(x')}$ = 0.56, which is the firing strength for A.
- *NSFLSs with centroid-composition based inference:* The centroid-based composition considers the area of the intersection between the input FS, I and the antecedent FS, A. The centroid of the intersection between I and A is defined in (3) [29],

$$x_{\text{cen}}(I \cap A) = \frac{\sum_{i=1}^{n} x_i \mu(x_i)}{\sum_{i=1}^{n} \mu(x_i)} \tag{3}$$

where n is the no. of discretizations in the intersection between I and A. The degree of membership of the centroid in the antecedent FS ($x_{cen}(I \cap A)$) is used as the firing strength, defined in (4).

$$\mu_{I \cap A}(x_{cen}(I \cap A) \tag{4}$$

In this case, Fig. 3 shows that the centroid of the intersection between I and A is $x_{\text{cen}}(I \cap A) = 1.85$ for $n = 100$. It intersects A at 'Q' with $\mu_{Cen-ns}(x')$ = 0.5 which serves as the firing strength for A.

The next section will discuss the development of SFLS-based and NSFLS-based movie recommendation systems along with the demonstration of accuracy improvement achieved through the latter system for noisy inputs.

III. SFLSs and NSFLSs Movie Recommendation

A SFLS-based movie recommendation system is proposed in [4] for the noise-free MovieLens data set [5]. If external noise is injected while processing this data set, it transforms this to a noisy set. It is expected that with such a data set, the performance of the SFLS-based system will degrade. We use the noisy data set in this paper as a proxy for vague and uncertainty movie ratings which could be elicited, e.g. via an interval-valued response mode [32] and platforms such as DECSYS [33]. An NSFLS-based movie recommendation is developed for this noisy data set to recommend movies to users, and its recommendation performance is assessed against the SFLS-based model in [4]. Figure 4 presents the entire procedure for movie recommendation systems with the SFLS or NSFLS.

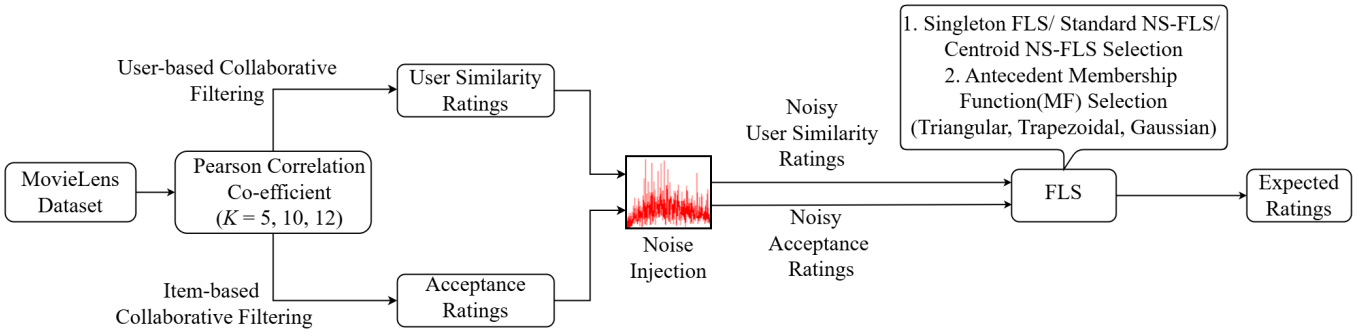


Fig. 4: Working procedure of a movie recommendation system using SFLS and NSFLS

A. Input Processing for Movie Recommendation System

The Movielens [5] data set is one of the widely used movie ratings data set. To prepare the input for the FLS, user-based and item-based CFs are performed. It is noted that the CF focuses only on user-item interactions and does not require detailed metadata about the items; in contrast, the CBF relies heavily on the attributes of the items [34]. In this process, we first search for users having tastes similar to a target user using the Pearson correlation coefficient [35], involving three types of *K*-neighbourhood values, $K = 5, 10,$ and 20. Note that using different types of *K*-neighbourhood values increases the control over bias and variance trade-off. The lower value of *K* leads to low bias and high variance, making the model more sensitive to noise or outliers, whereas an increase in the value of *K* results in an increase in bias and makes it robust to noise or outliers [36]. Equation (5) presents the Pearson Correlation Coefficient [35].

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \cdot \sum(y_i - \bar{y})^2}} \tag{5}$$

where x_i and y_i are data points. $\bar{x}$ and $\bar{y}$ are the means of x and y, respectively. r is Pearson correlation coefficient, ranging from -1 to $+1$.

With the user-based CF, we get one of the inputs for the FLS—'User Similarity Ratings'. The item-based CF identifies items similar to those with which the target user has liked or interacted. It uses the same *K*-neighborhood setup and similarity calculation (see (5)), resulting in the other input 'Acceptance Ratings' for the movie recommendation system.

B. Noise Generation and Injection

In this stage, we inject noise into the generated inputs. Generally, the amount of noise is described by the Signal-to-Noise ratio (SNR), measured in decibels (dB) [11]. If the noise level is low, the SNR value will be high. On the other hand, a lower SNR value indicates a high level of noise [37]. So, a 5dB SNR produces more noise to the data set than a 20dB SNR. Equation (6) defines how to calculate the SNR [31].

$$\text{SNR} = 10 \log_{10} \left(\frac{1}{\sigma_n^2} \right). \tag{6}$$

Here, σ_n serves two purposes. Firstly, it is used to generate and inject Gaussian noise into the data set. We set the mean value to 0 and σ_n is used as the standard deviation of the Gaussian distribution for the particular level of noise associated with the SNR. A random noise value is propagated from the calculated distribution and is added to the noise-free inputs for the FLS-based movie recommendation system. If the inputs—user similarity ratings and acceptance ratings are u_r and a_r, respectively, the noise is injected into them using (7).

$$\begin{aligned} u'_r &= u_r + \mathcal{N}(0, \sigma_n^2) \\ a'_r &= a_r + \mathcal{N}(0, \sigma_n^2) \end{aligned} \tag{7}$$

Here, u'_r and a'_r are the noisy user similarity ratings and noisy acceptance ratings respectively.

Secondly, σ_n is used to determine the standard deviation of the Gaussian input fuzzy set (FS) for the NSFLS, corresponding to a specific level of noise. For instance, when injecting 5dB of noise into the data set, the input FS for NSFLS is prepared with a standard deviation of 0.562, as calculated using (6). This ensures that the NSFLS can effectively handle the increased level of uncertainty associated with the noise. To systematically evaluate the system's performance under varying noise conditions, we gradually inject four levels of SNR: 5dB, 10dB, 15dB, and 20dB. The corresponding σ_n of the input FSs are 0.562, 0.316, 0.178, and 0.1, respectively (see Fig. 5, recreated from [31]).

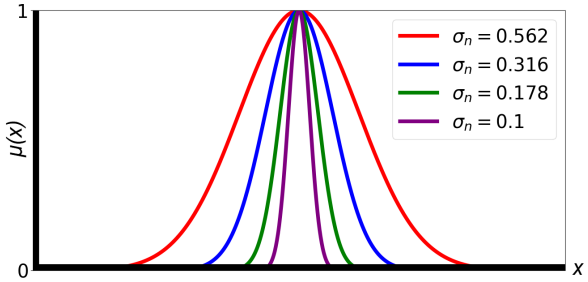
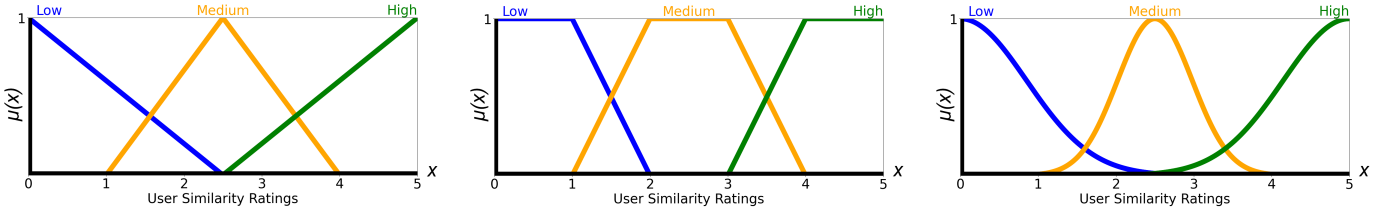


Fig. 5: Different standard deviation (σ_n) of input FSs for an NSFLS-based movie recommendation

C. Movie Recommendation FLS Process

After generating the noisy inputs, we select the FLS for the movie recommendation system. Here, three types of FLS are considered–SFLS as well as NSFLS with standard (Std)



(a) Triangular singleton Fuzzy Set (b) Trapezoidal singleton Fuzzy Set (c) Gaussian singleton Fuzzy Set

Fig. 6: Different types of antecedent fuzzy MF for movie recommendation FLS, recreated from [4]

and centroid (Cen) based composition. For every FLS, the movie recommendation system is designed with three types of antecedent MF types—triangular (Fig. 6a), trapezoidal (Fig. 6b), and Gaussian (Fig. 6c) using nine different if-then fuzzy rules obtained from [4]. In the case of non-singleton FSs, we use the Gaussian distribution for input FS with σ_n calculated with (6) according to the injected noise level. We compare the movie recommendation results among the designed SFLS, Std-NSFLS and Cen-NSFLS systems in respect to the Root Mean Sqaure Error (RMSE) as defined in (8).

IV. Result Analysis

The experiments are carried out in the MovieLens data set comprising one million movie ratings from 6000 users in 4000 movies where the ratings are given on a scale from 0 to 5 [5]. The initial data set is noise-free, but converted it to a noisy data set by gradually injecting 5dB, 10dB, 15dB, and 20dB of noise. For experiment, we consider different K-neighbourhood values such as 5, 10, 20 and MFs including triangular, trapezoidal and Gaussian. It is noted that through these experiments we demonstrate the differences in performance of the SFLS- and NSFLS(s)-based recommendation systems under different levels of input noise. All experiment results are presented in Table I. For performance evaluation, we use RMS that estimates the recommendation error by considering the average deviation of the predicted ratings from the observed ones [38], defined in (8)

$$RMSE = \sqrt{\frac{\Sigma(P_i - O_i)^2}{n}} \tag{8}$$

where P_i and O_i are the predicted and observed ratings, respectively and n is the total number of movie ratings. The lower the RMSE value, the higher the accuracy of the system.

TABLE I: Comparison of movie recommendation performance among SFLS, Std-NSFLS and Cen-NSFLS with respect to RMSE for MovieLens data set [5] with various noise levels considering different K-neighbourhood values and MFs.

MFs	K	Movie Recommendation FLS		
		SFLS	Std-NSFLS	Cen-NSFLS
Triangular	5	1.352	1.304	**1.291**
	10	1.453	1.374	**1.373**
	20	1.36	1.293	**1.291**
Trapezoidal	5	1.379	1.3	**1.294**
	10	1.5	**1.381**	1.39
	20	1.413	**1.294**	1.298
Gaussian	5	1.388	1.286	**1.284**
	10	1.507	1.387	**1.386**
	20	1.407	**1.298**	1.299

(a) With 5dB Injected Noise

MFs	K	Movie Recommendation FLS		
		SFLS	Std-NSFLS	Cen-NSFLS
Triangular	5	1.134	**1.126**	1.131
	10	1.165	**1.133**	1.137
	20	1.08	**1.038**	1.041
Trapezoidal	5	1.147	**1.127**	1.136
	10	1.201	**1.138**	1.139
	20	1.106	**1.052**	1.054
Gaussian	5	1.153	**1.123**	1.124
	10	1.204	**1.145**	1.15
	20	1.113	**1.048**	1.054

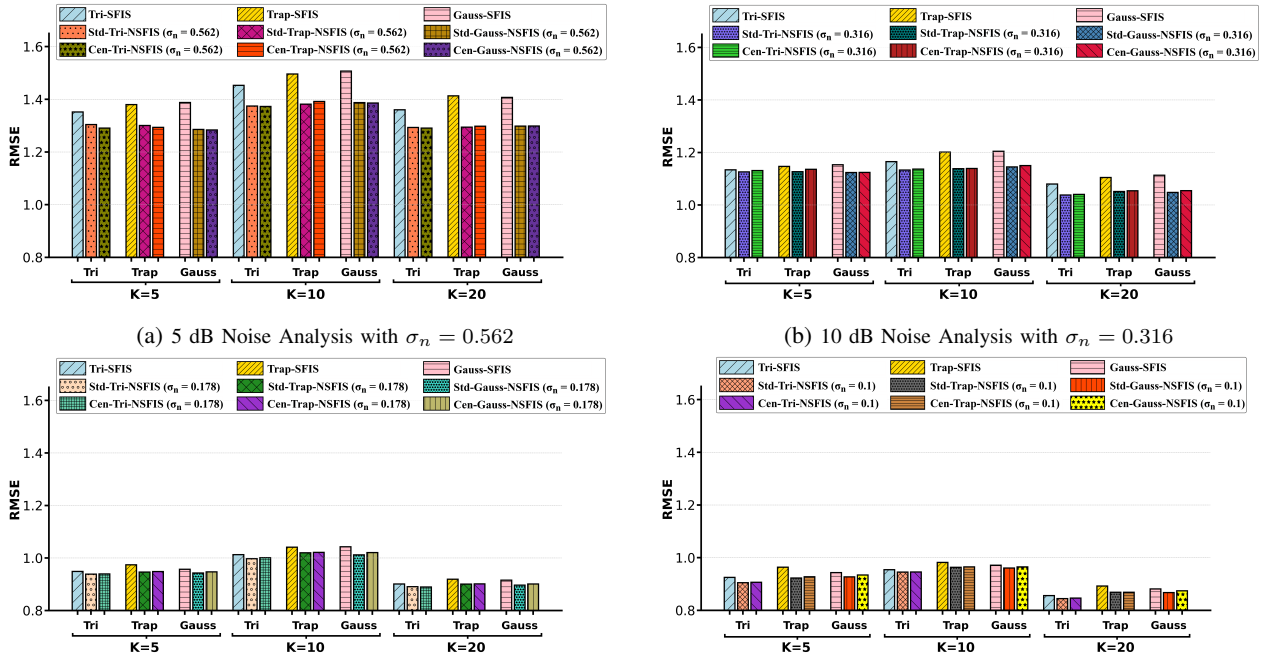
(b) With 10dB Injected Noise

MFs	K	Movie Recommendation FLS		
		SFLS	Std-NSFLS	Cen-NSFLS
Trapezoidal	5	0.948	**0.938**	0.939
	10	1.012	**0.997**	1.001
	20	0.9	0.891	**0.889**
Triangular	5	0.973	**0.946**	0.948
	10	1.041	**1.019**	1.021
	20	0.919	**0.9**	0.901
Gaussian	5	0.956	**0.942**	0.947
	10	1.042	**1.011**	1.021
	20	0.915	**0.896**	0.901

(c) With 15dB Injected Noise

MFs	K	Movie Recommendation FLS		
		SFLS	Std-NSFLS	Cen-NSFLS
Triangular	5	0.925	**0.904**	0.906
	10	0.954	**0.945**	0.946
	20	0.856	**0.844**	0.846
Trapezoidal	5	0.964	**0.922**	0.927
	10	0.981	**0.964**	0.965
	20	0.892	**0.868**	**0.868**
Gaussian	5	0.943	**0.927**	0.934
	10	0.971	**0.96**	0.965
	20	0.881	**0.867**	0.874

(d) With 20dB Injected Noise



(a) 5 dB Noise Analysis with $\sigma_n = 0.562$

(b) 10 dB Noise Analysis with $\sigma_n = 0.316$

(c) 15 dB Noise Analysis with $\sigma_n = 0.178$

(d) 20 dB Noise Analysis with $\sigma_n = 0.1$

Tri = Triangular, **Trap = Trapezoidal, **Gauss** = Gaussian*

Fig. 7: Comparing the *RMSE* among SFLS, Std-NSFLS and Cen-NSFLS for Different *K*-neighbourhood values and MFs

1) With 5dB Noise Injection: It is the most noisy condition in the experiment, as a lower SNR value indicates a higher noise. For triangular FSs, the Cen-NSFLS performs better than both SFLS and Std-NSFLS. The performance gap between Cen-NSFLS and Std-NSFLS is marginal, but there is a substantial difference in the RMSE value between SFLS and both NSFLSs shown in the Table-I(a). In the case of trapezoidal MFs, Std-NSFLS sightly outperforms Cen-NSFLS, where for Gaussian MFs, both Std-NSFLS and Cen-NSFLS consistently outperform SFLS across all MFs and *K*-neighborhood sizes. This indicates the effectiveness of non-singleton FS in handling input uncertainty in neighborhood-based movie recommendation systems.

2) With 10dB Noise Injection: For all types of MFs and *K*-neighbourhood values, Std-NSFLS consistently outperforms the other two systems, achieving the lowest RMSE value in every case. This highlights the effectiveness of NSFLSs in improving the accuracy of this application under input uncertainty. Cen-NSFLS performs slightly worse than Std-NSFLS, but is still better than SFLS in most cases. When used with NSFLSs, the Gaussian MF outperforms the other MFs, indicating that it better captures the underlying patterns in the dataset. While Cen-NSFLS generally performs better than SFLS shown in Table-I(b), it does not outperform Std-NSFLS, suggesting that using a Cen-NSFLS does not provide additional advantages for this particular case.

3) With 15dB Noise Injection: With this noisy case, Std-NSFLS achieves the best performance in most MFs, except for Trapezoidal MF at *K*=20, where Cen-NSFLS shows slightly better performance. Cen-NSFLS consistently performs close to that of Std-NSFLS and outperforms SFLS in most cases, suggesting it to be a strong alternative. From Table-I(c), it is visible that the Gaussian MF consistently provides the best performance across most configurations, making it the most effective MF for this noisy data set.

4) With 20dB Noise Injection: It is the least noisy condition for the MovieLens data set among the conditions used for the experiments. In this case, Std-NSFLS consistently achieves the lowest RMSE scores, showing its superiority in handling uncertainty and improving recommendation accuracy across all configurations. Under different levels of noise, it is clearly evident that NSFLSs outperform SFLS with lower *RMSE* as a movie recommendation system. In terms of analyzing noisy MovieLens data sets, the Std-NSFLS demonstrates better performance than the Cen-NSFLS in most cases; but their recommendation performances are very close, suggesting to use any of them depending on the noisy data set and application.

V. Conclusion

The main contribution of this paper focuses on the use of NSFLSs to improve the performance of movie recommendation systems in noisy situations. We discuss the challenges for existing approaches, in particular, the SFLS-based movie recommendation system, to deal with noisy input data. Illustrating the challenge with a comparative experiment, we demonstrate the difference in performance of the SFLS- and NSFLS(s)-based recommendation systems under different levels of input noise. The experimental results demonstrate that NSFLSs are

a useful approach for making movie recommendations with higher accuracy under noisy ratings. A limitation of the paper is that all noise is synthetic, i.e. no real-world noisy data was used. In the future, we expect to focus on vague, for example, interval-valued data sets, and to refine the NSFLS system to automatically adapt to varying levels of input noise as in [11]. We also want to continue the investigation of applying NSFLSs with optimized time complexity to address more complex real-world problems frequently facing input uncertainties.

REFERENCES

[1] M. Goyani and N. Chaurasiya, "A review of movie recommendation system: Limitations, survey and challenges," *ELCVIA: electronic letters on computer vision and image analysis*, vol. 19, no. 3, pp. 0018–37, 2020.

[2] N. Idrissi and A. Zellou, "A systematic literature review of sparsity issues in recommender systems," *Social Network Analysis and Mining*, vol. 10, no. 1, p. 15, 2020.

[3] M. Singh, "Scalability and sparsity issues in recommender datasets: a survey," *Knowledge and Information Systems*, vol. 62, no. 1, pp. 1–43, 2020.

[4] M. R. Siddiquee, N. Haider, and R. M. Rahman, "Movie recommendation system based on fuzzy inference system and adaptive neuro fuzzy inference system," *International Journal of Fuzzy System Applications*, vol. 4, no. 4, pp. 31–69, 2015.

[5] GroupLens, "MovieLens 1M Dataset," retrieved 2014-09-09. [Online]. Available: https://grouplens.org/datasets/movielens/

[6] P. S. Amartya, S. Kabir, S. C. Babu, and M. Jahan, "An interval creation approach to construct interval type-2 fuzzy sets," in *2024 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2024, pp. 1–9.

[7] J. H. Aladi, C. Wagner, J. M. Garibaldi, and A. Pourabdollah, "On transitioning from type-1 to interval type-2 fuzzy logic systems," in *IEEE International Conference on Fuzzy Systems*. IEEE, 2015, pp. 1–8.

[8] Y. Deldjoo, Z. He, J. McAuley, A. Korikov, S. Sanner, A. Ramisa, R. Vidal, M. Sathiamoorthy, A. Kasirzadeh, and S. Milano, "A review of modern recommender systems using generative models (gen-recsys)," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 6448–6458.

[9] D. Pekaslan, S. Kabir, J. M. Garibaldi, and C. Wagner, "Determining firing strengths through a novel similarity measure to enhance uncertainty handling in non-singleton fuzzy logic systems." in *IJCCI*, 2017, pp. 83–90.

[10] J. M. Mendel, "Non-singleton fuzzification made simpler," *Information Sciences*, vol. 559, pp. 286–308, 2021.

[11] D. Pekaslan, C. Wagner, and J. Garibaldi, "Adonis - adaptive online non-singleton fuzzy logic systems," *IEEE Transactions on Fuzzy Systems*, vol. PP, pp. 1–1, 08 2019.

[12] J. H. Aladi, C. Wagner, A. Pourabdollah, and J. M. Garibaldi, "Contrasting singleton type-1 and interval type-2 non-singleton type-1 fuzzy logic systems," in *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2016, pp. 2043–2050.

[13] Z. Su, H. Yang, and J. Ai, "Fplv: Enhancing recommender systems with fuzzy preference, vector similarity, and user community for rating prediction," *Plos one*, vol. 18, no. 8, p. e0290622, 2023.

[14] N. Pradeep, K. Rao Mangalore, B. Rajpal, N. Prasad, and R. Shastri, "Content based movie recommendation system," *International journal of research in industrial engineering*, vol. 9, no. 4, pp. 337–348, 2020.

[15] H. A. Basha, S. B. Sangeetha, S. Sasikumar, J. Arunnehru, and M. Subramaniam, "A proficient video recommendation framework using hybrid fuzzy c means clustering and kullback-leibler divergence algorithms," *Multimedia Tools and Applications*, vol. 82, no. 14, pp. 20 989–21 004, 2023.

[16] M. Gupta, A. Thakkar, V. Gupta, D. P. S. Rathore *et al.*, "Movie recommender system using collaborative filtering," in *2020 international conference on electronics and sustainable communication systems (ICESC)*. IEEE, 2020, pp. 415–420.

[17] T. Anwar and V. Uma, "Comparative study of recommender system approaches and movie recommendation using collaborative filtering," *International Journal of System Assurance Engineering and Management*, vol. 12, pp. 426–436, 2021.

[18] N. Khairova, N. Sharonova, D. Sytnikov, M. Hrebeniuk, and P. Sytnikova, "Recommendation system based on a compact hybrid user model using fuzzy logic algorithms." in *COLINS (2)*, 2024, pp. 48–62.

[19] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[20] Z. Batmaz, A. Yurekli, A. Bilge, and C. Kaleli, "A review on deep learning for recommender systems: challenges and remedies," *Artificial Intelligence Review*, vol. 52, pp. 1–37, 2019.

[21] D. Mudigere, Y. Hao, J. Huang, Z. Jia, A. Tulloch, S. Sridharan, X. Liu, M. Ozdal, J. Nie, J. Park *et al.*, "Software-hardware co-design for fast and scalable training of deep learning recommendation models," in *Proceedings of the 49th Annual International Symposium on Computer Architecture*, 2022, pp. 993–1011.

[22] J. M. Alonso Moral, C. Castiello, L. Magdalena, C. Mencar, J. M. Alonso Moral, C. Castiello, L. Magdalena, and C. Mencar, "Designing interpretable fuzzy systems," *Explainable Fuzzy Systems: Paving the Way from Interpretable Fuzzy Systems to Explainable AI Systems*, pp. 119–168, 2021.

[23] L. Magdalena, "Fuzzy systems interpretability: What, why and how," *Fuzzy Approaches for Soft Computing and Approximate Reasoning: Theories and Applications: Dedicated to Bernadette Bouchon-Meunier*, pp. 111–122, 2021.

[24] C. Wagner, S. Miller, J. M. Garibaldi, D. T. Anderson, and T. C. Havens, "From interval-valued data to general type-2 fuzzy sets," *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 2, pp. 248–269, 2014.

[25] M. Tang and H. Liao, "From conventional group decision making to large-scale group decision making: What are the challenges and how to meet them in big data era? a state-of-the-art survey," *Omega*, vol. 100, p. 102141, 2021.

[26] A. Jain and C. Gupta, "Fuzzy logic in recommender systems," *Fuzzy Logic Augmentation of Neural and Optimization Algorithms: Theoretical Aspects and Real Applications*, pp. 255–273, 2018.

[27] M. M. R. Siddiquee, N. Haider, and R. M. Rahman, "A fuzzy based recommendation system with collaborative filtering," in *The 8th International Conference on Software, Knowledge, Information Management and Applications (SKIMA 2014)*. IEEE, 2014, pp. 1–8.

[28] H. D. Calderon Vilca, "Netflix movie recommendation using fuzzy logic," 01 2022.

[29] A. Pourabdollah, C. Wagner, and J. Aladi, "Changes under the hood-a new type of non-singleton fuzzy logic system," in *2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2015, pp. 1–8.

[30] C. Chen, Y. Zhao, C. Wagner, D. Pekaslan, and J. M. Garibaldi, "An extension of the fuzzyr toolbox for non-singleton fuzzy logic systems," in *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2021, pp. 1–6.

[31] C. Fu, A. Sarabakha, E. Kayacan, C. Wagner, R. John, and J. M. Garibaldi, "A comparative study on the control of quadcopter uavs by using singleton and non-singleton fuzzy logic controllers," in *2016 IEEE international conference on fuzzy systems (FUZZ-IEEE)*. IEEE, 2016, pp. 1023–1030.

[32] Z. Ellerby, C. Wagner, and S. B. Broomell, "Capturing richer information: On establishing the validity of an interval-valued survey response mode," *Behavior Research Methods*, pp. 1–23, 2021.

[33] Z. Ellerby, J. McCulloch, J. Young, and C. Wagner, "Decsys–discrete and ellipse-based response capture system," in *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2019, pp. 1–6.

[34] S. A. Thorat, G. Ashwini, and M. Seema, "Survey on collaborative and content-based recommendation systems," in *2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT)*. IEEE, 2023, pp. 1541–1548.

[35] I. Cohen, Y. Huang, J. Chen, J. Benesty, J. Benesty, J. Chen, Y. Huang, and I. Cohen, "Pearson correlation coefficient," *Noise reduction in speech processing*, pp. 1–4, 2009.

[36] M. P. Boobalan, D. Lopez, and X. Z. Gao, "Graph clustering using k-neighbourhood attribute structural similarity," *Applied soft computing*, vol. 47, pp. 216–223, 2016.

[37] D. Pekaslan, C. Wagner, and J. M. Garibaldi, "Adonis—adaptive online nonsingleton fuzzy logic systems," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 10, pp. 2302–2312, 2019.

[38] S. Kabir, C. Wagner, and Z. Ellerby, "Toward handling uncertainty-at-source in ai—a review and next steps for interval regression," *IEEE Transactions on Artificial Intelligence*, vol. 5, no. 1, pp. 3–22, 2024.