International Journal of Computational Intelligence Systems
Vol. **12(2)**, 2019, *pp.* 1382–1392
DOI: https://doi.org/10.2991/ijcis.d.191115.001; ISSN: 1875-6891; eISSN: 1875-6883
https://www.atlantis-press.com/journals/ijcis/

Exploring Fuzzy Rating Regularities for Managing Natural Noise in Collaborative Recommendation

Raciel Yera[1], Manuel J. Barranco[2,*], Ahmad A. Alzahrani[3], Luis Martínez[2]

[1]*University of Ciego de Ávila, Carretera a Morón Km. 9 1/2, Ciego de Ávila, Cuba*
[2]*Computer Science Department, University of Jaén, Campus Las Lagunillas, 23071, Jaén, Spain*
[3]*Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, 21589, Saudi Arabia*

ARTICLE INFO

Article History
Received 27 Sep 2019
Accepted 09 Nov 2019

Keywords
Recommender systems
Natural noise
Regularities
Fuzzy logic

ABSTRACT

Recommender systems have played a relevant role in e-commerce for supporting online users to obtain suggestions about products that best fit their preferences and needs in overloaded search spaces. In such a context, several authors have proposed methods focused on removing the users' inconsistencies when they rate items, so-called natural noise, improving in this way the recommendation performance. The current paper explores the use of rating regularities for managing the natural noise in collaborative filtering recommendation, having as key feature the use of fuzzy techniques for coping with the uncertainty associated to such scenarios. Specifically, such regularities are used for representing common rating patterns and thus detect noisy ratings when they tend to contradict such patterns. An experimental study is developed for showing the performance of the proposal, as well as analyzing its behavior in contrast to previous natural noise management procedures.

1. INTRODUCTION

Recommender systems (RSs) have become an outstanding tool for providing personalized information about items (e.g. movies, books, e-services) in overloaded search spaces [1,2]. In such spaces, the huge amount of items make difficult to users the task of choosing the most suitable items according to their current preferences and needs. RSs are then highly appreciated for facilitating the access to the right information for each individual user.

Different paradigms have driven the development of RSs in which two of them outstand over the others: (i) content-based approaches [3], focused on suggesting items with similar features to those preferred in the past by the current user; and (ii) collaborative filtering approaches [4,5], focused on suggesting items preferred in the past by other users which have similar preferences to the current user. Beyond these paradigms, several authors have referred to other paradigms such as social, knowledge-based or hybrid filtering [6], taking into account the information sources and techniques that were used to generate the desired recommendations.

The majority of the recommendation approaches assume that the user ratings are free of inconsistencies, and are then focused on proposing new methods centerd on directly improving the recommendation accuracy. However, some recent research works reveal that such ratings can be either inconsistent or noisy, and it has been pointed out the existence of a "magic barrier," which can limit the reaching of recommendation improvements due to such rating inconsistencies [7]. Several authors have shown that these inconsistencies can be caused by users' personal conditions, social influences, emotional states, contexts, or certain rating scales [8]. Due to the fact that they appear without a malicious or premeditated intention, such inconsistencies have been coined as *natural noise* by the research community.

Natural noise management has then become a key aspect to improve the performance of RSs. In this way, the research works in this area can be divided in two groups: (i) those that need additional information beyond the rating values to perform the noise management [9–12], and (ii) those that are able to perform the noise management using as input only the preferences values [13–15]. While there are several research efforts associated to the first group, there are still few research works belonging to the second one.

This work is focused on the proposal of a new approach for natural noise management that takes as initial input only the user preference values, and retrieves a de-noised dataset that leads to the improvement of the recommendation performance. Furthermore, it manages the uncertainty associated to the recommendation scenario through the use of fuzzy concepts. With this goal in mind, the approach introduces the concept of rating regularity. Even though, this concept has not had a common use in RSs research, it is closely related with frequent itemsets and association rules in the RS scenario [16–18], which have had a wider application by the research community.

*Corresponding author. Email: barranco@ujaen.es

Specifically, the main contributions of the paper consist of:

1. Introducing the rating regularity concept in the collaborative filtering scenario, as a tool for capturing common behaviors that could be useful in the detection of anomalous rating patterns that could result in natural noise.
2. Introducing a fuzzy transformation for the rating values as well as for rating regularities, that takes into account the uncertainty associated to the RS scenario.
3. Presenting an approach for noise degree calculation, based on the identified regularities after their fuzzy transformation.
4. Validate the proposal through studying the sensitivity of their main parameters, and comparing with regards to previous research.

The paper is structured as follows. Section 2 presents the necessary background for the current proposal presentation. Section 3 presents the proposal, which includes the formalization of rating regularities, regularities detection, regularities filtering, noise degree calculation, and noise detection and correction. Section 4 presents a case study for validating the proposal. Section 5 concludes the paper.

2. BACKGROUND

This section provides the necessary background for the proposal presentation, which includes basic notions about RSs, some related works on natural noise management and elementary concepts of fuzzy sets (FSs) theory.

2.1. Recommender Systems

RSs are considered as "any system that produces individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options" [19]. In a similar direction, Gunawardana and Shani [20] have pointed out that the two more common tasks related to RSs are the prediction task (i.e. the prediction of the user ratings about a group of items), and the recommendation task (i.e. the recommendation of a set of interesting and useful items to the user). Based on such goals, several recommendation approaches have been developed. One of the most popular classification of RSs has been provided by Bobadilla *et al.* [6], which groups them into a) demographic filtering, b) collaborative filtering, c) content-based filtering and d) hybrid filtering. Specifically, since 90s the collaborative and the content-based filtering have played a relevant role both at research-oriented and at application-oriented scenarios.

The most widely used paradigm for developing recommendation approaches has been the collaborative filtering, that is focused on performing the prediction and the recommendation tasks through only users' rating values [1]. This paradigm usually generates the recommendations for the current user, by exploring the preferences of other related users regarding their degree of similarity. Such an exploration is typically based on their rating patterns. This approach does not depend on items attributes; therefore it could be used in any recommendation scenario having enough preference values.

On the other hand, content-based RSs use the item's descriptions and a profile with the interests of the active user, for suggesting items similar to those the user already preferred in the past [3]. Content-based recommendation focuses on comparing the user profile and the candidate items, to find the items that should be suggested. Items profiles are usually represented through a set of attributes that can include weights to represent the importance of each one of them [1]. The user profiles are then represented by aggregating the profiles associated to their preferred items.

2.2. Related Works on Natural Noise Management in Recommender Systems

The majority of the recommendation approaches assume that the user ratings are free of inconsistencies, and then are focused on proposing new methods centerd on directly improving the recommendation accuracy. However, some recent research works reveal that such ratings can be either inconsistent or noisy [14,21]. Such noise has been grouped in two main categories according to the purpose of the user introducing erroneous information in the system [14]: (i) malicious noise, which is intentionally introduced by users to bias the recommendation and promote/demote certain products or diminish the system quality [21], or (ii) natural noise, which is introduced by users without malicious intentions when they provide their preferences [14]. Specifically, natural noise has attracted the attention of the researchers in the last few years, as inconsistencies that can be caused by users' personal conditions, social influences, emotional states, contexts, or certain rating scales [8].

On the other hand, recent researches have shown that there is a "magic barrier" in recommendation performance that algorithms were reaching, and it prevents them from improving their results [7]. In order to overcome such magic barrier it is necessary to check several elements in the input of the recommendation algorithm, such as the rating scale, or inconsistencies in preferences [9]. With this regard, natural noise management is focused on mitigating the negative influence of inconsistent preferences in the RSs performance [9,14,22].

Several research works have been focused on natural noise management in RSs. As it was pointed out in the Introduction section, these works can be divided in two groups: (i) those that depend on additional information for performing the natural noise management (e.g. item attributes, semantic information), and (ii) those that are able to perform the noise management using as input only the preferences values. Table 1 presents such two groups of recent related works, and introduces a new classification based on the use of crisp or fuzzy techniques for information modeling in each analyzed work.

Regarding the research works that need additional information for performing its role, most of them depend on additional information that could be difficult to obtain in certain scenarios, and therefore lack of generalization capacity. Here, Amatriain *et al.* [9] proposed the mining and usage of a de-noised dataset with information provided by experts to reduce noise. Pham and Jung [11] used item attributes to create user models and correct the ratings that do not match such models, built by using information of other users identified as experts. In a different direction, Said and Bellogin

Table 1 Research works focused on natural noise management.

	Additional Information	Only Ratings
Crisp	Amatriain *et al.* [9]	
	Pham and Jung [11]	Li *et al.* [13]
	Saia *et al.* [12]	Yera *et al.* [14]
	Said and Bellogin [7]	Bag *et al.* [23]
	Dixit *et al.* [10]	
Fuzzy		Yera *et al.* [15]
		Moses and Babu [24]

[7] used item attributes for measuring user coherence in the rating patterns, showing that the recommendation accuracy is improved when the less coherent users are discarded from the dataset. Saia *et al.* [12] also presented an approach for removing incoherent items from a user profile, using semantic information. Finally, Dixit *et al.* [10] have brought the natural noise management into the context-aware recommendations, by proposing in this scenario a framework for noise detection and correction that depends on the contextual dimensions beyond the user-item matrix.

In a lesser extent, some authors have recently developed approaches focused on natural noise management using only the rating values. Li *et al.* [13] proposed the discovery of noisy but non-malicious users by detecting user's self-contradictions, regarding that highly-correlated items should receive similar rating value. Here the authors are centered on noise management at the user level by considering the removal of top-noisy rating for improving the recommendation accuracy. To manage noise at the rating level, Yera *et al.* [14] proposed a method for correcting noisy preferences following the principle that users and items have their own tendency giving or receiving ratings. Once the tendencies have been identified, the ratings that contradict them can be classified as possibly noisy and corrected by performing a new rating prediction for the same user and item. Recently, Bag *et al.* [23] followed a similar idea for natural noise management in highly sparse scenarios.

Furthermore, we have also detected a small group of works that consider the management of uncertainty associated to rating values through the use of fuzzy logic, and only rely on rating values for natural noise management. Here, Yera *et al.* [15] used fuzzy tools for composing user, item, and rating profiles; and identified as noisy to the ratings where the corresponding user and item profiles are close enough, but far from the rating profile. Noisy ratings are corrected through the prediction of a new rating value for the same user and item using a traditional collaborative filtering algorithm. Following this scheme, Moses and Babu [24] have also proposed a noise detection algorithm that formalizes the use of a fuzzy linguistic approach.

The analysis of the related literature concludes that there are few works focused on natural noise management using fuzzy techniques in spite of their potential to improve natural noise management. In addition, we have detected that the developed works suffer from some drawbacks such an important intrusiveness level and a high computational cost because they have embedded a collaborative filtering algorithm. The current research work aims at mitigating such drawbacks by proposing a new approach for natural noise management that uses the concept of rating regularity and also incorporates the uncertainty management through fuzzy techniques.

2.3. Fuzzy Sets Theory: Basic Concepts

This section reviews briefly different basic concepts of FSs theory such as, FS, fuzzy number (FN), linguistic variable, and so on that will be used in the main proposal of this research work.

Definition 1. [25] FSs extend the notion of a set by introducing the degree of membership of elements. This establishes a correspondence between the elements of the universe of discourse X into the interval $[0, 1]$, which is given by a membership function:

$$\mu_{\tilde{A}} : X \rightarrow [0, 1] \tag{1}$$

A FS $\tilde{A}$ on X is represented by a set of pairs of elements $x \in X$ and its membership degree:

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \in X\} \tag{2}$$

Definition 2. [26] A FN is a FS $\tilde{A}$ on $\mathbb{R}$ that satisfies two conditions:

- *Normality*: There exists at least one number $x \in \mathbb{R}$ whose membership value is one, i.e. $\mu_{\tilde{A}}(x) = 1$.
- *Convexity*: $\forall x, y \in \mathbb{R}$ and $\forall \lambda \in [0, 1]$ we have

$$\mu_{\tilde{A}}(\lambda x + (1 - \lambda) y) \geqslant min\{\mu_{\tilde{A}}(x), \mu_{\tilde{A}}(y)\}$$

Definition 3. [26] A **triangular FN** $\tilde{A}$ is a FN characterized by the membership function

$$\mu_{\tilde{A}}(x) = \begin{cases} 0 & t < a \\ \dfrac{t-a}{b-a} & a \leqslant t < b \\ \dfrac{c-t}{c-b} & b < t \leqslant c \\ 0 & t > c \end{cases} \tag{3}$$

A triangular FN is given by a tuple (a, b, c), where the base of the triangle is the interval $[a, c]$ and the vertex is at $x = b$.

Taking as basis these concepts, the use of linguistic descriptors based on the fuzzy linguistic approach [27], has been a straightforward and popular tool to model the uncertainty and vagueness inherent to the human reasoning and natural language. Using words or linguistic descriptors, humans are able to valuate some subjective aspects, rather than using numbers. The concept of linguistic variable arises to support this reasoning, where values are not numbers but words.

Definition 4. [27] A **linguistic variable** V is characterized by a quintuple (V, T, X, G, M) where:

- V is the name of the variable
- T is the terms set of V, i.e. the set of linguistic values of V
- X is the universe of discourse
- G is a syntactic grammar that produces the linguistic values
- M is a semantic rule which associates a subset of X to each terms of T.

Typically, triangular FNs are used to provide the semantic rule in the context of RSs [28,29]. Figure 1 shows an example of a linguistic variable, using triangular FNs, being the terms set

$$T = \{nothing, very\ bad, bad, medium, good, very\ good, perfect\}$$

and the semantic rule

$$\begin{aligned} M(nothing) &= (0, 0, 0.17) \\ M(very\ bad) &= (0, 0.17, 0.33) \\ M(bad) &= (0.17, 0.33, 0.5) \\ M(medium) &= (0.33, 0.5, 0.67) \\ M(good) &= (0.5, 0.67, 0.83) \\ M(very\ good) &= (0.67, 0.83, 1) \\ M(perfect) &= (0.83, 1, 1) \end{aligned}$$

3. USING FUZZY RATING REGULARITIES FOR NATURAL NOISE MANAGEMENT

Here, it is presenting our new proposal that uses fuzzy rating regularities for natural noise management in RSs. Initially the concept of rating regularity is formalized (Section 3.1). Subsequently, the four stages of the proposal are presented (Figure 2): regularities detection (Section 3.2), regularities filtering (Section 3.3), noise degree calculation (Section 3.4), and noise detection and correction (Section 3.5).

Furthermore, Figure 2 shows the role of this proposal in a collaborative filtering scenario, as an alternative to remove noise from the rating data before the application of the collaborative filtering recommendation approach.

To facilitate the proposal presentation, Table 2 contains the main notation used across the paper.

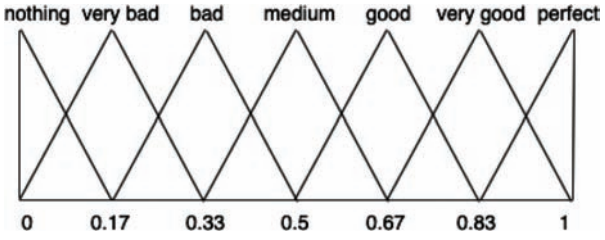

Figure 1 Semantic of a linguistic variable using triangular fuzzy numbers (FNs).

3.1. Formalizing Rating Regularities

The definition of regularity takes as base the concept of frequent itemsets, widely used in association rule mining [30]. Consider $I = \{i_1, i_2, i_3, ...\}$ as a set of n items, and $D = \{t_1, t_2, t_3, ...\}$ as a set of transactions that is considered as the database, where each transaction in D contains a subset of the items in I. In such scenario, a frequent itemset is a subset X of the set of items I, $X \subset I$. The presence of a frequent itemset can be interpreted as a set of items that co-occur across many transactions in the database D. This concept has been used by several authors in RS researches in order to represent the users' preferences, for guaranteeing an effective and transparent recommendation generation [16–18,31].

Based on the concept of frequent itemset, in this work we propose the concept of regularity as a frequent itemset but not composed by simple items. Instead, it is composed by a term (called regularity term), that represents a possible rating value of the current user over certain item. As far as we know, we only identified the use of the concept of regularity in RSs in the research work developed by Yera *et al.* [32], where the authors presented some evidences that regularities could be used for performing data preprocessing in RSs. However, such work only presents an initial analysis and does not consider any kind of uncertainty management.

Table 2 Main notation used across the proposal.

Notation	Meaning
r_{ui}	Rating of user u over item i
val_{ui}	Value of the rating r_{ui} in a regularity term
$r_{ui} = val_{ui}$	Regularity term
Reg_u	Rating regularity composed of a set of regularity terms
S	Set of regularities Reg_u
S^*	Set of regularities Reg_u after the filtering process
$\tilde{r}_{ui}$	Fuzzy transformation of r_{ui}
$\widetilde{Reg}_u$	Fuzzy transformation of the regularity Reg_u
$\mu_{high}(r_{ui})$	Membership value of r_{ui} to the set *high*
$\mu_{medium}(r_{ui})$	Membership value of r_{ui} to the set *medium*
$\mu_{low}(r_{ui})$	Membership value of r_{ui} to the set *low*
$\tilde{a}_{ui}$	Fuzzy transformation of the rating to get its noise degree
$deg(\tilde{a}_{ui})$	Noise degree of $\tilde{a}_{ui}$

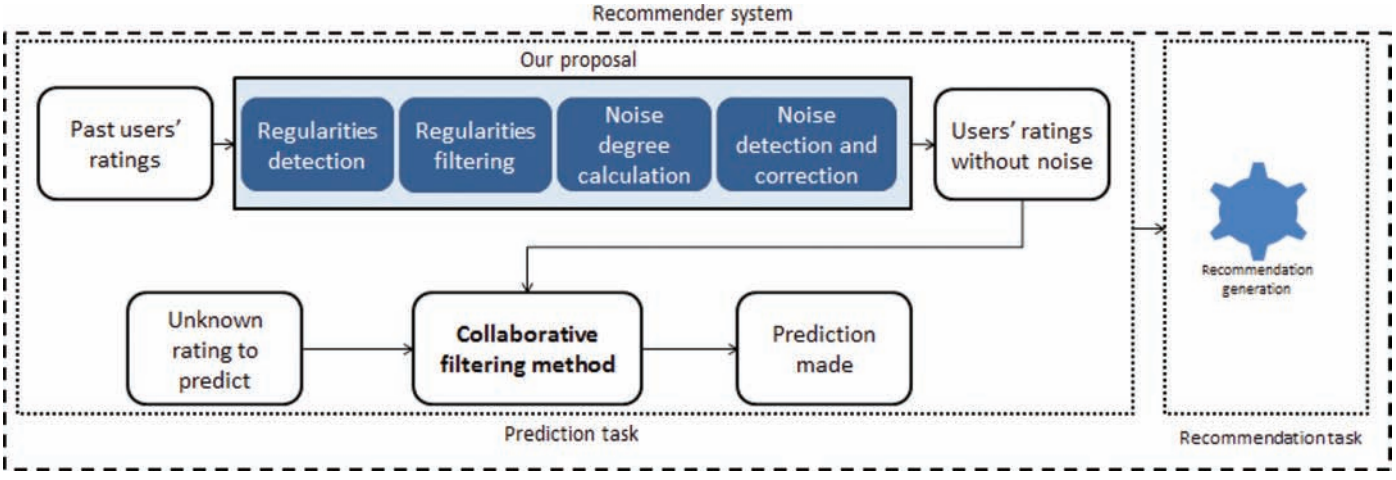

Figure 2 Scheme of the proposed approach in a collaborative filtering scenario.

To define the regularity concept, at first it is necessary to formally define the concept of *regularity term*.

Definition 5. A regularity term is a term with the form $r_{ui} = val_{ui}$, where r_{ui} represents the rating provided by the user u over the item i, and val_{ui} is a value of the rating scale associated to the current recommendation domain which in this case is the value associated to r_{ui}.

Example 1. In order to show a demonstrative example across this proposal presentation, Table 3 shows a small dataset of a collaborative filtering RS with 4 users and 4 items. Here $r_{ui_1} = 5$ is a regularity term associated to a user u over the item i_1, and has the value $val_{ui_1} = 5$.

A regularity is then defined as follows:

Definition 6. A regularity Reg_u is defined as a set of regularity terms associated to the same user u, but over different items i.

Therefore, a user satisfies a regularity if he/she satisfies all the regularity terms associated to such regularity. This leads to the definition of support.

Definition 7. The support of a regularity Reg_u is the amount of users u that satisfy such regularity.

Example 2. Regarding the mentioned example in Table 3, the regularity $\{r_{ui_1} = 5, r_{ui_4} = 2\}$ has a *support* = 3, because there are 3 users (i.e. u_2, u_3, and u_4) that satisfy it.

3.2. Regularities Detection

We then consider each user's set of ratings as an individual transaction composed by a set of regularity terms that represent the ratings of the current user (e.g. according to Table 3, u_1 would be represented as $< r_{ui_1} = 5, r_{ui_2} = 4, r_{ui_3} = 3, r_{ui_4} = 1 >$). Taking into account this representation, it could be used a traditional algorithm for frequent itemset discovery for finding the users' regularities [30,33].

Here, we use an Apriori-like algorithm to find such regularities [30]. The Apriori algorithm comprises the detection of frequent itemsets in transactional data, and is composed by two stages. The first stage of the algorithm simply counts item occurrences to determine the large 1-itemsets. The second stage is composed by k passes, where in the pass k the large itemsets L_{k-1} found in the $(k-1)$ *th* pass are employed for generating the candidate itemsets that are filtered to composed the current large itemsets L_k.

Algorithm 1 presents an overview of this method, contextualized to the current scenario. At first, lines 01–03 initialize the three auxiliary sets 1 – *sized* – *regularities*, *large* – *regularities*, and *final* – *set*. Subsequently, lines 04–09 present the first stage of the algorithm, which scans the rating data and builds all the regularities composed of only one regularity term, which support is greater than or equal to the minimum support value received as input. These regularities are added to the set *final* – *set* that will be used at the end of the algorithm to retrieve the final-set of regularities, and also added to the set *large* – *regularities*, that is used in the second stage of the algorithm.

Table 3 Working scenario for the proposal. Ratings in range [1; 5].

	i_1	i_2	i_3	i_4
u_1	5	4	3	1
u_2	5	5	3	2
u_3	5	4	3	2
u_4	5	3	3	2

Algorithm 1: Algorithm for regularities detection

Input: *R*-Set of ratings r_{ui}. *sup*-Minimum support
Output: *S*-Set of regularities Reg_u
01: 1-sized-regularities={}
02: large-regularities={}
03: final-set={}
04: For each item *i* and each possible rating value val_{ui}
05: Count = the amount of users satisfying $r_{ui} = val_{ui}$
06: If Count $\geqslant$ sup
07: $Reg_u = \{r_{ui} = val_{ui}\}$
08: 1-sized-regularities.Add(Reg_u)
09: large-regularities.Add(Reg_u)
10: While large-regularities != {}
11: final-set.add(large-regularities)
12: New-regularity-set={}
13: For each regularity Reg_a in large-regularities
14: For each regularity Reg_b in1-sized-regularities
15: If Reg_b is not a subset of Reg_a
16: $Reg_{new} = Reg_a \cup Reg_b$
17: Count = the amount of users satisfying Reg_{new}
18: If ($Count \geqslant sup$)
19: New-regularity-set.Add(Reg_{new})
20: large-regularities=New-regularity-set
21: End-While
22: Return final-set as *S*

The second stage (lines 10–21), is composed of an iterative procedure where the iteration k generates all the available regularities containing exactly $k + 1$ regularity terms. This generation is done by combining all the regularities obtained at the iteration $k - 1$ (having size k), with all the regularities with size 1 (lines 13–14). Once the union between both regularities is performed, it is checked whether the new regularity satisfies the minimum support condition (line 18). In the positive case, it is added to the current set of regularities. The procedure finishes where certain iteration n was not able to discover any regularity of size $n + 1$. We remark that Algorithm 1 is based on a global scheme of Apriori [30]. Several authors have also proposed implementation tricks for executing this scheme in faster way, but they are not presented at this paper considering its actual scope [33,34].

Algorithm 1 uses as parameter the minimum support for considering a regularity as valid (i.e. a regularity which support is under such minimum value, is not discovered by the algorithm). Such parameter will be adjusted in the experimental section.

3.3. Regularities Filtering

It is expected that the set of the discovered regularities using the Apriori-like algorithm, has similar regularities and therefore contains redundancy. Considering Table 3 and a minimum support = 3, some valid regularities are:

$$Reg_u^1 = \left\{r_{ui_1} = 5\right\} \tag{4}$$

$$Reg_u^2 = \left\{r_{ui_1} = 5, r_{ui_3} = 3\right\} \tag{5}$$

$$Reg_u^3 = \left\{r_{ui_1} = 5, r_{ui_3} = 3, r_{ui_4} = 2\right\} \tag{6}$$

However, both Reg_u^1 and Reg_u^2 are subsets of Reg_u^3, and then the set of regularities contains repetitive information in this way.

To avoid such redundancy, we include a filtering stage that removes any regularity which is a subset of other regularity also included in the set. Algorithm 2 shows this procedure.

Algorithm 2: Regularity filtering algorithm

Input: S-Set of regularities
Output: S^*-Set of filtered regularities
1: For each pair (Reg_a, Reg_b) in S
2: If (Reg_a) is a subset of (Reg_b)
3: Remove (Reg_a) from S
4: Return S as S^*

3.4. Noise Degree Calculation

Once the set of filtered regularities is obtained, such set is used as the key information source to calculate the noise degree of a given rating.

The idea of noise degree calculation consists of analyzing the ratings of those items that belong to a regularity in the filtered set and assign a greater noise degree when such rating values do not match with the values of the regularity. To manage the inherent uncertainty of this process, we use fuzzy modeling for our information representation.

We then consider the transformation of the rating values into a fuzzy representation according to their membership degree to three FSs that represent their tendency to be *high*, *medium*, or *low* preferences.

Definition 8. The fuzzy representation, $\tilde{r}_{ui}$, of a rating value, r_{ui}, is defined by its membership degree to three FSs, as a three dimensional vector:

$$\tilde{r}_{ui} = \left(\mu_{high}\left(r_{ui}\right), \mu_{medium}\left(r_{ui}\right), \mu_{low}\left(r_{ui}\right)\right) \tag{7}$$

being high, medium, and low, FSs associated to the rating scale domain and represented by the fuzzy membership functions showed at Figure 3, in which the value a plays a key role for defining the three FSs as a fuzzy partition, meanwhile *min* and *max* are the minimum and maximum values of the rating scale.

The fuzzy representation of the ratings leads to the definition of a fuzzy regularity.

Definition 9. A fuzzy regularity, $\widetilde{Reg}_u$, consists of the fuzzy representation of each rating value associated to the regularity terms, r_{ui}, in the regularity Reg_u (see Eq. 8).

$$\begin{aligned}\widetilde{Reg}_u = \{&\tilde{r}_{ui_1} = \left(\mu_{high}\left(val_{ui_1}\right), \mu_{medium}\left(val_{ui_1}\right), \mu_{low}\left(val_{ui_1}\right)\right),\\ &\tilde{r}_{ui_2} = \left(\mu_{high}\left(val_{ui_2}\right), \mu_{medium}\left(val_{ui_2}\right), \mu_{low}\left(val_{ui_2}\right), ...\right\}\end{aligned} \tag{8}$$

Algorithm 3 presents the procedure to calculate the noise degree of a rating, based on these concepts. Once the fuzzy regularities and the fuzzy representation of the ratings have been obtained, the noise degree of a rating r_{ui} is calculated by the sum of the distances between its fuzzy representation $\tilde{r}_{ui}$ and the fuzzy representation of the regularity terms located at the detected regularities $\widetilde{Reg}_u$.

Algorithm 3: Algorithm for noise degree calculation

Input: $\tilde{a}_{ui}$-Fuzzy representation of the rating to calculate its noise degree. S^*-Set of identified regularities, after its fuzzy transformation
Output: $deg\left(\tilde{a}_{ui}\right)$-Noise degree of $\tilde{a}_{ui}$
1: $deg\left(\tilde{a}_{ui}\right)$=0
2: For each regularity $\widetilde{Reg}_u$ in S^*
3: If $(\widetilde{Reg}_u)$ has a regularity term $\tilde{r}_{ui}$= $\widetilde{val}_{ui}$ referencing the item i also associated to $\tilde{a}_{ui}$
4: $deg\left(\tilde{a}_{ui}\right) = deg\left(\tilde{a}_{ui}\right) + \text{dist}(\tilde{a}_{ui}, \widetilde{val}_{ui})$
5: Return $deg\left(\tilde{a}_{ui}\right)$

Initially the noise degree value is initialized as 0 (line 1). Afterward, line 2 at Algorithm 3 checks all the discovered regularities, considering their fuzzy transformation. For each one, line 3 checks whether the current fuzzy regularity has a term that makes reference to the same item associated to the rating received as input to calculate its noise; i.e. for the input rating $\tilde{a}_{ui}$, it is checked whether the item i is also in some of the regularity terms associated to the current regularity $\widetilde{Reg}_u$. In the positive case, the noise degree value is incremented with the distance between the fuzzy representation of such input, and the fuzzy representation of the value in the regularity term (line 4). Specifically, being $\tilde{r}_{ui} = \widetilde{val}_{ui}$ the regularity term in the fuzzy regularity $\widetilde{Reg}_u$ which item i is the same item at the input rating $\tilde{a}_{ui}$, the noise degree is then increased as the distance between $\tilde{a}_{ui}$ and $\widetilde{val}_{ui}$. Eventually, the noise degree will be the sum of the distances between the input rating and all the terms associated to each corresponding regularity. At the end of the algorithm, such calculated degree is returned (line 5).

Figure 4 also shows a visual representation of this process, which is a key component of the whole approach for natural noise management.

This algorithm required the use of a distance for calculating the dissimilarity between two fuzzy-transformed ratings. In this case it will be applied the Manhattan distance between such three dimensional

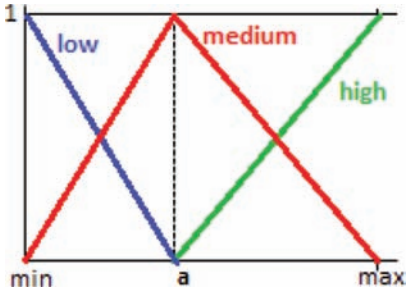

Figure 3 | Membership function of the fuzzy sets low, medium, and high.

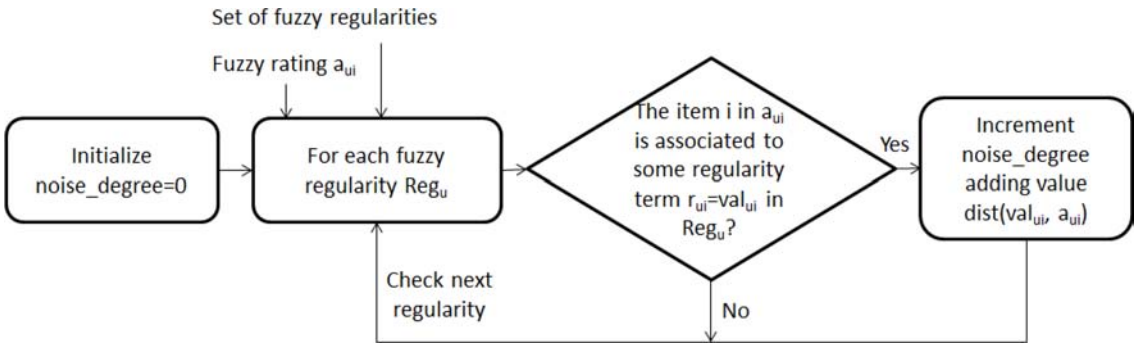

Figure 4 | Scheme of noise degree calculation process.

vectors, regarding that its use in similar scenarios of fuzzy profiling in RSs has been previously discussed and justified [15]. Such distance reflects in a direct way the differences between the membership values, and considers the differences between the dimensions without giving importance to its distribution accross the dimensions.

The Manhattan distance between two n-dimensional vectors $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ is calculated as [35]:

$$dist(x, y) = \sum_{i=1}^{n} |x_i - y_i| \tag{9}$$

Being $\tilde{a}_{ui}$ and $\tilde{r}_{ui}$ two fuzzy representations of rating values, the Manhattan distance can be calculated as:

$$\begin{aligned} dist(\tilde{a}_{ui}, \tilde{r}_{ui}) = & |\mu_{high}(a_{ui}) - \mu_{high}(r_{ui})| + \\ & |\mu_{medium}(a_{ui}) - \mu_{medium}(r_{ui})| + \\ & |\mu_{low}(a_{ui}) - \mu_{low}(r_{ui})| \end{aligned} \tag{10}$$

Previous works on a different scenario have shown that the value $dist(\tilde{a}_{ui}, \tilde{r}_{ui})$ lies in the range $[0, 2]$ [15].

Example 3 presents an example of a fuzzy regularity as well as its use in the whole noise degree calculation process that is currently being presented.

Example 3. Assume Table 3 where the only one regularity, being the minimum support = 3, is $Reg_u^3 = \{r_{ui_1} = 5, r_{ui_3} = 3, r_{ui_4} = 2\}$, and the parameters of the membership functions in Figure 3, defined as: $min = 1$, $max = 5$, and $a = 3$.

To obtain the noise degree of $r_{u_1 i_4}$, it would be done as follows:

1. The fuzzy transformation of Reg_u^3 would be as follow: $\widetilde{Reg}_u^3 = \{\tilde{r}_{ui_1} = (1, 0, 0), \tilde{r}_{ui_3} = (0, 1, 0), \tilde{r}_{ui_4} = (0, 0.5, 0.5)\}$
2. The fuzzy transformation of $r_{u_1 i_4} = 1$ would be as follows: $\tilde{a}_{u_1 i_4} = (0, 0, 1)$.
3. Directly executing the Algorithm 3; in this case there is only one regularity.
4. Therefore, in such iteration, it is satisfied the condition at line 3, because such regularity contains a term $(\tilde{r}_{ui_4} = (0, 0.5, 0.5))$ which item (i_4) matches with the item at the rating which noise degree is required to calculate $(\tilde{a}_{u_1 i_4} = (0, 0, 1))$.
5. Afterward, the noise degree is calculated as the distance between both fuzzy representations (see Eq. 10). This is:
$$\begin{aligned} & deg(\tilde{a}_{u_1 i_4}) = \text{dist}\left(\tilde{a}_{u_1 i_4}, \widetilde{val}_{ui_4}\right) = \\ & |\mu_{high}(a_{u_1 i_4}) - \mu_{high}(val_{ui_4})| + |\mu_{medium}(a_{u_1 i_4}) - \\ & \mu_{medium}(val_{ui_4})| + |\mu_{low}(a_{u_1 i_4}) - \mu_{low}(val_{ui_4})| \\ & = |0 - 0| + |0 - 0.5| + |1 - 0.5| = 1. \end{aligned}$$

3.5. Noise Detection and Correction

Once the noise degrees have been calculated, we process as noisy the top-n ratings with higher noise degree for each user. This research work will take into account several values for the parameter n, in order to consider a lower or higher intrusiveness degree.

Even though, previous works consider sophisticated rating replacement strategies that incorporate the use of collaborative filtering schemes to predict rating values [14,15], this work will consider simpler strategies with a lower computational cost. Specifically, it will consider two different strategies that can be applied indistinctly, depending on the desired goal of reaching a more or less accuracy improvement by removing or modifying ratings. The strategies are:

1. Remove the rating detected as noisy.
2. Replace the noisy ratings with the average rating associated to the current user.

Future works will consider regularities-driven strategies for noise correction in the framework associated to this research work (Figure 2).

Example 4. Assume the working scenario presented in the Example 1, i.e. Table 3 where the only one regularity is $Reg_u^3 = \{r_{ui_1} = 5, r_{ui_3} = 3, r_{ui_4} = 2\}$ and rating $a_{u_1 i_4} = 1$. Such rating was already analyzed at Example 3, concluding that it has a noise degree $deg(\tilde{a}_{u_1 i_4}) = 1$. Considering $n = 1$, the rating $r_{u_1 i_4}$ is the top-noisy rating for u_1, and therefore it is necessary to correct it. Here the replacement strategy # 1 performs the rating removal, implying that u_1 would not have a rating value for the item i_4. On the other hand, the strategy # 2 would calculate a new value $r_{u_1 i_4} = 3.25$ because it is the average rating for the user u_1.

4. CASE STUDY

This section provides a case study to show how the application of the proposal as a preprocessing stage in the rating data, leads to an improvement in the recommendation accuracy. At first the experimental setup will be presented, including datasets and evaluation protocol. Furthermore, the experimental results are presented and discussed.

Datasets The proposal will be evaluated using three well-known datasets in RS research.

- Movielens 100K, composed of 943 users and 1682 items, with a sparsity of 0.9369.
- MovieTweeting, specifically it will be used a subset composed of 1692 users and 10123 items, with a sparsity of 0.995.
- Netflix, specifically it will be used a subset composed of 1758 users and 1001 items, with a sparsity of 0.974.

Evaluation protocol The datasets are prepared through the approach presented by Gunawardana and Shani [20]. To build training and test sets, they choose a set of users from the dataset and randomly hide n ratings for each user, where n is also randomly selected for each user. Such hidden ratings are used to build the test set, and the remaining ones are then the training set. This partitioning process is performed several times and the results are averaged. Taking as base these training and test sets, the proposal is evaluated according to this procedure:

1. To apply the current proposal over the training set, obtaining the modified training set.
2. To recommend with a given recommendation method using the modified training set, which is a de-noised version of the original training set. In this case we will use the item-based collaborative filtering approach, which is a well-recognized and widespread recommendation approach [36]. Here the number of neighbors in the prediction calculation are fixed to $k = 60$.
3. To evaluate the recommendation results using an evaluation measure. In this case we use the Mean Absolute Error (MAE) [20], which focuses on how well the recommendation technique predicts the hidden ratings.
4. To compare the evaluation results against the recommendation results by using the original training set, instead of the modified one.

This proposal also depends on the parameter a to set in the membership functions at the FSs low, medium, and high. In the scope of this paper, we will take the more generalized scenario and assign $a = (min + max)/2$; being min and max the minimum and maximum rating for the current recommendation domain. In future works we will consider other criteria for representing the uncertainty associated to the specific context. On the other hand, the minimum support and the top-n ratings per user to process as noisy are studied in this experimental analysis.

Analysis of the results Tables 4–9 present the MAE results of the execution of the proposals for the three datasets. Specifically, the tables show the results associated to the two noise correction strategies (the average and the removal strategies), considering a minimum support value in the range [20; 70]. Furthermore, it was also considered the value n in the top-n ratings per user to process as noisy, in the range [1; 10], even though in some scenarios other values of n were considered.

At first, it is significant that for the six experimental scenarios, the proposal leads to improvements in the recommendation performance in relation to the baseline values (specified in the caption of the corresponding tables). Here we recall that the baseline values are represented by the recommendation performance without considering the application of the proposal for natural noise management.

In the case of the natural noise management using the average user rating as correction strategy the proposal clearly outperforms such baseline reaching a MAE value of 0.7650 (baseline 0.7705) for Movielens; 1.1814 (baseline 1.2076) for MovieTweeting; and 0.7821 (baseline 0.8012) for Netflix.

It is also remarkable that for Movielens and MovieTweeting such results were obtained just for n = 3 and n = 2 respectively (i.e. modify 3 and 2 ratings per user), indicating that only the transformation of a small portion of the user profile, can lead to a relevant improvement in the recommendation performance.

In this way, for all the minimum support values, the tendency was to decrease the recommendation accuracy with the increment of n. This fact suggests that a higher intrusiveness degree in the user profile using the proposed correction strategies, seems to introduce noise, instead of removing it. In relation to the minimum support values, we also report a tendency to decrease the recommendation accuracy with the increasing of such support. Such fact suggests that a higher minimum support value gives the reaching of too general regularities that do not cover more specific knowledge that is useful for the noise correction. We also remark that we do not present results for a minimum support lower than 20 because the computational cost of the process disables its reaching in a rational time.

On the other hand, in the case of the removal strategy for noise correction, the results were most modest in relation to the average strategy. Here it was obtained a MAE value of 0.7673 (baseline 0.7705) for Movielens; 1.1933 (baseline 1.2076) for MovieTweeting; and 0.8006 (baseline 0.8012) for Netflix. For all cases, such results were obtained for n = 1 (i.e. modifying only 1 ratings per user). Furthermore, for higher values of n it was obtained a performance worse that the baseline. This behavior was expected, regarding that the information losing notably affects any recommendation scenario. However, it is notable that even a natural noise management approach that introduces information losing, can lead to recommendation improvements when such losing is slight and at very specific scenarios. In other direction, for the removal strategy the variation of the minimum support and the n values across their parameter scales, has a similar behavior in relation to the average strategy.

Comparison with previous works: In this subsection we compare our proposal against the most relevant previous work in natural noise management, according to our criteria in relation to this contribution. This is the proposal developed by Yera *et al.* [15], that is one of the few works on natural noise management that uses

Table 4 Results for the dataset Movielens. Average strategy. Baseline 0.7705.

Support/n	1	2	3	4	5	6	7	8	9	10
20	0.7671	0.7668	0.7653	0.7659	0.7657	0.7679	0.7694	0.7716	0.7725	0.7741
30	0.7669	0.7663	**0.7650**	0.7655	0.7670	0.7677	0.7686	0.7707	0.7722	0.7729
40	0.7729	0.7667	0.7665	0.7666	0.7669	0.7679	0.7696	0.7707	0.7717	0.7725
50	0.7679	0.7667	0.7671	0.7669	0.7672	0.7669	0.7677	0.7685	0.7705	0.7717
60	0.7689	0.7681	0.7668	0.7668	0.7677	0.7696	0.7702	0.7711	0.7723	0.7733
70	0.7688	0.7692	0.7680	0.7686	0.7693	0.7707	0.7719	0.7733	0.7739	0.7749

Bold values indicate best results.

Table 5 Results for the dataset Movielens. Removal strategy. Baseline 0.7705.

Support/n	1	2	3	4	5	6	7	8	9	10
20	**0.7673**	0.7683	0.7690	0.7713	0.7694	0.7727	0.7741	0.7761	0.7778	0.7814
30	0.7681	0.7693	0.7691	0.7701	0.7728	0.7734	0.7742	0.7782	0.7798	0.7795
40	0.7688	0.7696	0.7722	0.7724	0.7731	0.7765	0.7783	0.7824	0.7855	0.7869
50	0.7692	0.7707	0.7739	0.7748	0.7758	0.7759	0.7780	0.7790	0.7815	0.7828
60	0.7707	0.7707	0.7715	0.7730	0.7727	0.7757	0.7792	0.7800	0.7809	0.7826
70	0.7702	0.7730	0.7720	0.7721	0.7734	0.7770	0.7798	0.7815	0.7820	0.7851

Bold values indicate best results.

Table 6 Results for the dataset MovieTweeting. Average strategy. Baseline 1.2076.

Support/n	1	2	3	4	5	6	7	8	9	10
20	1.1844	**1.1814**	1.1815	1.1839	1.1870	1.1887	1.1912	1.1927	1.1971	1.1991
30	1.1886	1.1897	1.1892	1.1887	1.1899	1.1915	1.1926	1.1927	1.1953	1.1978
40	1.1962	1.1925	1.1927	1.1920	1.1902	1.1916	1.1919	1.1947	1.1962	1.1985
50	1.1934	1.1923	1.1932	1.1918	1.1917	1.1925	1.1915	1.1929	1.1949	1.1992
60	1.1967	1.1905	1.1905	1.1918	1.1937	1.1951	1.1944	1.1945	1.1970	1.1981
70	1.1937	1.1926	1.1917	1.1944	1.1974	1.1966	1.1967	1.1980	1.2025	1.2037

Bold values indicate best results.

Table 7 Results for the dataset MovieTweeting. Removal strategy. Baseline 1.2076.

Support/n	1	2	3	4	5	6	7	8	9	10
20	**1.1933**	1.1945	1.1971	1.2029	1.2085	1.2137	1.2211	1.2218	1.2278	1.2290
30	1.1964	1.2029	1.2060	1.2099	1.2186	1.2236	1.2230	1.2340	1.2366	1.2419
40	1.2031	1.2080	1.2121	1.2191	1.2197	1.2245	1.2298	1.2345	1.2389	1.2426
50	1.2027	1.2092	1.2193	1.2212	1.2227	1.2238	1.2231	1.2247	1.2284	1.2342
60	1.2069	1.2121	1.2160	1.2183	1.2230	1.2263	1.2263	1.2267	1.2329	1.2351
70	1.2058	1.2128	1.2136	1.2217	1.2237	1.2277	1.2307	1.2335	1.2365	1.2370

Bold values indicate best results.

Table 8 Results for the dataset Netflix. Average strategy. Baseline 0.8012.

Support/n	1	2	3	4	5	6	7	8	9	10	11	12
20	0.7917	0.7887	0.7862	0.7864	0.7844	0.7846	0.7850	0.7845	0.7825	0.7830	0.7821	0.7831
30	0.7911	0.7891	0.7867	0.7862	0.7854	0.7861	0.7853	0.7843	0.7836	0.7827	0.7836	0.7841
40	0.7911	0.7902	0.7874	0.7852	0.7851	0.7837	0.7834	0.7833	0.7826	**0.7821**	0.7828	0.7831
50	0.7920	0.7897	0.7863	0.7849	0.7854	0.7854	0.7850	0.7850	0.7835	0.7838	0.7840	0.7835
60	0.7905	0.7884	0.7863	0.7871	0.7860	0.7858	0.7849	0.7846	0.7842	0.7852	0.7862	0.7857
70	0.7923	0.7884	0.7864	0.7863	0.7868	0.7865	0.7846	0.7839	0.7840	0.7846	0.7851	0.7846

Bold values indicate best results.

fuzzy tools for representing a user profile, an item profile, and a rating profile; and identifies as noisy to such ratings where the user and item profile are close enough, but far from the rating profile. Additionally, for noisy ratings it performs their correction by predicting a new rating value for the same user and item.

This last step adds an important computational complexity to the Yera *et al.* [15] proposal regarding it makes a new prediction for any detected noisy rating; in contrast to our current proposal which performs a very light prediction step that includes a simple average, and also identifies the required regularities in a very short time period regarding the nature of the RSs datasets.

Table 10 shows a comparison between our proposal and Yera *et al.* [15], in terms of recommendation accuracy (MAE) and also the amount of corrected ratings (it would be desirable a lower amount of corrected ratings to avoid intrusiveness). The table proves that even though our proposal has a lower computational cost in

Table 9 Results for the dataset Netflix. Removal strategy. Baseline 0.8012.

Support/n	1	2	3	4	5	6	7	8	9	10
20	0.8017	0.8041	0.8069	0.8067	0.8095	0.8100	0.8115	0.8140	0.8162	0.8142
30	0.8018	0.8042	0.8070	0.8048	0.8107	0.8133	0.8111	0.8117	0.8154	0.8164
40	0.8030	0.8055	0.8053	0.8035	0.8097	0.8084	0.8087	0.8102	0.8102	0.8105
50	0.8041	0.8044	0.8044	0.8031	0.8091	0.8116	0.8119	0.8157	0.8149	0.8175
60	0.8034	0.8041	0.8042	0.8067	0.8085	0.8095	0.8113	0.8133	0.8157	0.8189
70	0.8044	0.8016	0.8054	0.8060	0.8151	0.8127	0.8130	0.8160	0.8171	0.8227
80	**0.8006**	0.8011	0.8033	0.8079	0.8106	0.8094	0.8071	0.8100	0.8113	0.8160
90	0.8012	0.8034	0.8109	0.8155	0.8156	0.8128	0.8091	0.8154	0.8173	0.8185

Bold values indicate best results.

Table 10 Comparison of our proposal against a previous related research.

Method	MAE	Amount of Corrected Ratings
Movielens		
Yera *et al.* [15]	0.7655	8663
Our proposal (Average strategy)	**0.7650**	**2652**
MovieTweeting		
Yera *et al.* [15]	**1.1784**	3932
Our proposal (Average strategy)	1.1814	**3338**
Netflix		
Yera *et al.* [15]	**0.7820**	**3647**
Our proposal (Average strategy)	0.7821	13695

MAE, Mean Absolute Error. Bold values indicate best results.

relation to Yera *et al.* [15], it is competitive with this research work and even outperforms it in some scenarios. In the case of Movielens, our proposal outperforms Yera *et al.* [15] by getting a lower MAE value and even reaching it by needing the modification of a lower amount of ratings. For MovieTweeting, our proposal obtains its best accuracy value by needing the modification of a lower amount of ratings, and even thought does not achieve the value at Yera *et al.*, widely improves the original baseline and then could be used in several scenarios that require a low intrusiveness. Finally, for Netflix both methods obtain a similar recommendation accuracy, but our research work required the modification of a higher amount of ratings. This finding was obtained because the best result was at $n = 10$, needing the modification of a high amount of ratings for each user. However, in addition to Table 10 we point out that for lower values of n such as $n = 3$ and $support = 20$, the approach obtained a MAE value of 0.7862 modifying only 5167 ratings, and therefore reaching in this way competitive results also for the Netflix dataset.

5. CONCLUSIONS

The current research work introduces the use of fuzzy rating regularities in the context of the natural noise management task. With this goal in mind, it proposes an approach composed of four stages which are regularities detection, regularities filtering, noise degree calculation, and noise detection and correction. The working principle of the proposal is that the regularities represent common rating behaviors that can be used to identify as noise to such ratings that contradict them.

The experimental results of the evaluation of the proposal conclude that it outperforms the baseline for all the considered datasets and experimental scenarios. Specifically, the best results tend to be obtained for lower values of the minimum support and lower values of the top-n noisy ratings corrected for each user. Furthermore, it was proven that the proposal is competitive with a recent related work in the state of art, outperforming it in several scenarios.

Future works will be focused on: 1) exploring the behavior of the proposal in the group recommendation scenario [37]; 2) incorporating the time dimension into the proposal, and 3) exploring its effect in the recommendation performance at more specific recommendation domains, such as an e-learning scenario [38].

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHORS' CONTRIBUTIONS

R. Yera, M. Barranco and A.A Alzahrani participate in the development of the proposal, its evaluation, discussion of results, and paper writing. L. Martínez participates in the development of the proposal, and guides the discussion of the results and paper writing.

ACKNOWLEDGMENTS

This work was supported by the Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah, under grant No. (DF-679-611-1441). The authors, therefore, gratefully acknowledge DSR technical and financial support.

REFERENCES

[1] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions, IEEE Trans. Knowl. Data Eng. 17 (2005), 734–749.
[2] R. Yera, L. Martínez, Fuzzy tools in recommender systems: a survey, Int. J. Comput. Intell. Syst. 10 (2017), 776–803.
[3] P. Lops, M. De Gemmis, G. Semeraro, Content-based recommender systems: state of the art and trends, in: F. Ricci, L. Rokach, B. Shapira, P.B. Kantor (Eds.), Recommender Systems Handbook, Springer, Boston, 2011, pp. 73–105.
[4] M.D. Ekstrand, J.T. Riedl, J.A. Konstan, Collaborative filtering recommender systems, Found. Trends Hum. Comput. Interact. 4 (2010), 81–173.

[5] X. Su, T.M. Khoshgoftaar, A survey of collaborative filtering techniques, Adv. Artif. Intell. 2009 (2009), 1–19.
[6] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez, Recommender systems survey, Knowl. Based Syst. 46 (2013), 109–132. ISSN 0950-7051.
[7] A. Said A. Bellogin, Coherence and inconsistencies in rating behavior: estimating the magic barrier of recommender systems, User Model. User-Adap. Interact. 28 (2018), 97–125.
[8] D. Kluver, T.T. Nguyen, M. Ekstrand, S. Sen, J. Riedl, How many bits per rating?, in Proceedings of the Sixth ACM Conference on Recommender Systems, ACM, 2012, pp. 99-106.
[9] X. Amatriain, J.M. Pujol, N. Tintarev, N. Oliver, Rate it again: increasing recommendation accuracy by user re-rating, in Third ACM Conference on Recommender Systems, ACM, New York, USA, 2009, pp. 173–180.
[10] V.S. Dixit, P. Jain, S. Gupta, Proposed rcfs-cars framework with noise detection and correction, Appl. Artif. Intell. 33 (2019), 361–377.
[11] H.X. Pham, J.J. Jung, Preference-based user rating correction process for interactive recommendation systems, Multi. Tools Appl. 65 (2013), 119–132.
[12] R. Saia, L. Boratto, S. Carta, A semantic approach to remove incoherent items from a user profile and improve the accuracy of a recommender system, J. Intell. Inf. Syst. 47 (2016), 111–134.
[13] B. Li, L. Chen, X. Zhu, C. Zhang, Noisy but non-malicious user detection in social recommender systems, World Wide Web. 16 (2013), 677–699.
[14] R. Yera, Y. Caballero Mota, L. Martínez, Correcting noisy ratings in collaborative recommender systems, Knowl. Based Syst. 76 (2015), 96–108.
[15] R. Yera, J. Castro, L. Martínez, A fuzzy model for managing natural noise in recommender systems, Appl. Soft Comput. 40 (2016), 187–198.
[16] A. Ghoshal, S. Menon, S. Sarkar, Recommendations using information from multiple association rules: a probabilistic approach, Inf. Syst. Res. 26 (2015), 532–551.
[17] M. Kumara Swamy, P. Krishna Reddy, Improving diversity performance of association rule based recommender systems, in DEXA 2015, LNCS, Springer, Valencia, 2015, vol. 9261, pp. 499–508.
[18] W. Lin, S.A. Alvarez, C. Ruiz, Efficient adaptive-support association rule mining for recommender systems, Data Mining Knowl. Discov. 6 (2002), 83–105.
[19] R. Burke, Hybrid recommender systems: survey and experiments, User Model. User Adap. Interact. 12 (2002), 331–370.
[20] A. Gunawardana, G. Shani, A survey of accuracy evaluation metrics of recommendation tasks, J. Mach. Learn. Res. 10 (2009), 2935–2962.
[21] I. Gunes, C. Kaleli, A. Bilge, H. Polat, Shilling attacks against recommender systems: a comprehensive survey, Artif. Intell. Rev. 42 (2014), 767–799.
[22] L. Martínez, J. Castro, R. Yera, Managing natural noise in recommender systems, in Theory and Practice of Natural Computing: 5th International Conference, TPNC 2016, Springer International Publishing, Sendai, 2016, pp. 3–17.
[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, Decis. Support Syst. 118 (2019), 46–57.
[24] J.S. Moses, L.D. Babu, A fuzzy linguistic approach-based non-malicious noise detection algorithm for recommendation system, Int. J. Fuzzy Syst. 20 (2018), 2368–2382.
[25] L.A. Zadeh, Fuzzy sets, Inf. Control. 8 (1965), 338–353.
[26] D. Dubois, H. Prade, Operations on fuzzy numbers, Int. J. Syst. Sci. 9 (1978), 613–626.
[27] L. Zadeh, The concept of a linguistic variable and its applications to approximatereasoning-Part I, Inf. Sci. 8 (1975), 199–249.
[28] L. Martínez, L.G. Pérez, M. Barranco, A multigranular linguistic content-based recommendation model, Int. J. Intell. Syst. 22 (2007), 419–434.
[29] L. Martínez, M.J. Barranco, L.G. Perez, M. Espinilla, A knowledge-based recommender system with multigranular linguistic information, Int. J. Comput. Intell. Syst. 1 (2008), 225–236.
[30] R. Agrawal, R. Srikant, Fast algorithms for mining association rules, in 20th International ConferenceVery Large Data Bases, Santiago de Chile, 1994, pp. 487–499.
[31] C.W.-k. Leung, S.C.-f. Chan, F.-l. Chung, A collaborative filtering framework based on fuzzy association rules and multiple-level similarity, Knowl. Inf. Syst. 10 (2006), 357–381.
[32] R. Yera, Y. Caballero Mota, M. Garcia Borroto, A regularity-based preprocessing method for collaborative recommender systems, J. Inf. Process. Syst. 9 (2013), 435–460.
[33] C. Borgelt, Frequent item set mining, Wiley Interdiscip. Rev. Data Min. Knowl. Discov. 2 (2012), 437–456.
[34] C. Borgelt, R. Kruse, Induction of association rules: apriori implementation, in Compstat, Springer, Berlin, Germany, 2002, pp. 395–400.
[35] S.-H. Cha, Comprehensive survey on distance/similarity measures between probability density functions, Int. J. Math. Models Methods Appl. Sci. 1 (2007), 300–307.
[36] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in WWW, ACM, Hong Kong, 2001, pp. 285–295.
[37] J. Castro, R. Yera, L. Martínez, An empirical study of natural noise management in group recommendation systems, Decis. Support Syst. 94 (2017), 1–11.
[38] R. Yera, L. Martínez, A recommendation approach for programming online judges supported by data preprocessing techniques, Appl. Intell. 47 (2017), 277–290.