informatics

MDPI

Article

A Recommender System for Programming Online Judges Using Fuzzy Information Modeling

Raciel Yera Toledo [1], Yailé Caballero Mota [2] and Luis Martínez [3,*]

[1] Knowledge Management Center, University of Ciego de Ávila, Carretera a Morón Km 9½, 65100 Ciego de Ávila, Cuba; ryera@unica.cu
[2] Department of Computer Science, University of Camagüey, Circun. Km 5½, 70100 Camagüey, Cuba; yaile.caballero@reduc.edu.cu
[3] Department of Computer Science, University of Jaén, 23071 Jaén, Spain
* Correspondence: martin@ujaen.es

Received: 19 February 2018; Accepted: 29 March 2018; Published: 3 April 2018

Abstract: Programming online judges (POJs) are an emerging application scenario in e-learning recommendation areas. Specifically, they are e-learning tools usually used in programming practices for the automatic evaluation of source code developed by students when they are solving programming problems. Usually, they contain a large collection of such problems, to be solved by students at their own personalized pace. The more problems in the POJ the harder the selection of the right problem to solve according to previous users performance, causing information overload and a widespread discouragement. This paper presents a recommendation framework to mitigate this issue by suggesting problems to solve in programming online judges, through the use of fuzzy tools which manage the uncertainty related to this scenario. The evaluation of the proposal uses real data obtained from a programming online judge, and shows that the new approach improves previous recommendation strategies which do not consider uncertainty management in the programming online judge scenarios. Specifically, the best results were obtained for short recommendation lists.

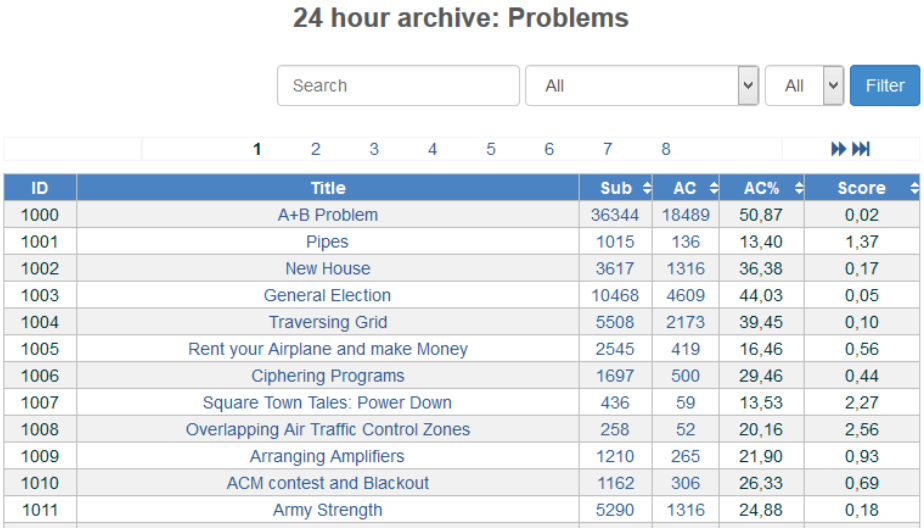
Keywords: programming online judges; fuzzy logic; problems recommendation

1. Introduction

Programming online judges (POJs) are e-learning platforms which have been widely accepted in the last few years for programming practices in computer science education and for training in competitive programming scenarios [1,2]. Basically, they are composed of a collection of programming exercises to be completed by the users. As an important feature, these platforms automate the evaluation of the proposed solution by doing an instantaneous verdict regarding its correctness. Key examples of POJs are the University of Valladolid Online Judge with more than 210,000 users and 1700 problems, and the Peking University Online Judge with more than 600,000 users and 3000 problems.

The typical interaction sequence between the user and the POJ is as follows:

- The user selects a problem to solve from the online judge (Figure 1).
- The code for solving such a problem is written.
- This source code is uploaded to the online judge (Figure 2).
- The judge automatically indicates whether the solution is correct or not. If not, classifies the failure mainly in Wrong Answer, Time limited exceeded or Runtime Error, allowing new solution attempts to the same problem.



ID	Title	Sub	AC	AC%	Score
1000	A+B Problem	36344	18489	50,87	0,02
1001	Pipes	1015	136	13,40	1,37
1002	New House	3617	1316	36,38	0,17
1003	General Election	10468	4609	44,03	0,05
1004	Traversing Grid	5508	2173	39,45	0,10
1005	Rent your Airplane and make Money	2545	419	16,46	0,56
1006	Ciphering Programs	1697	500	29,46	0,44
1007	Square Town Tales: Power Down	436	59	13,53	2,27
1008	Overlapping Air Traffic Control Zones	258	52	20,16	2,56
1009	Arranging Amplifiers	1210	265	21,90	0,93
1010	ACM contest and Blackout	1162	306	26,33	0,69
1011	Army Strength	5290	1316	24,88	0,18

Figure 1. Typical interface with the set of problems to solve in an online judge.

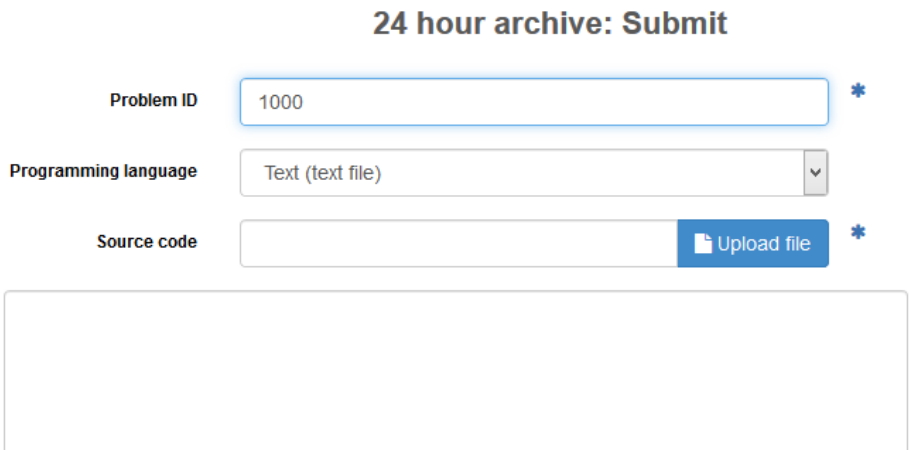


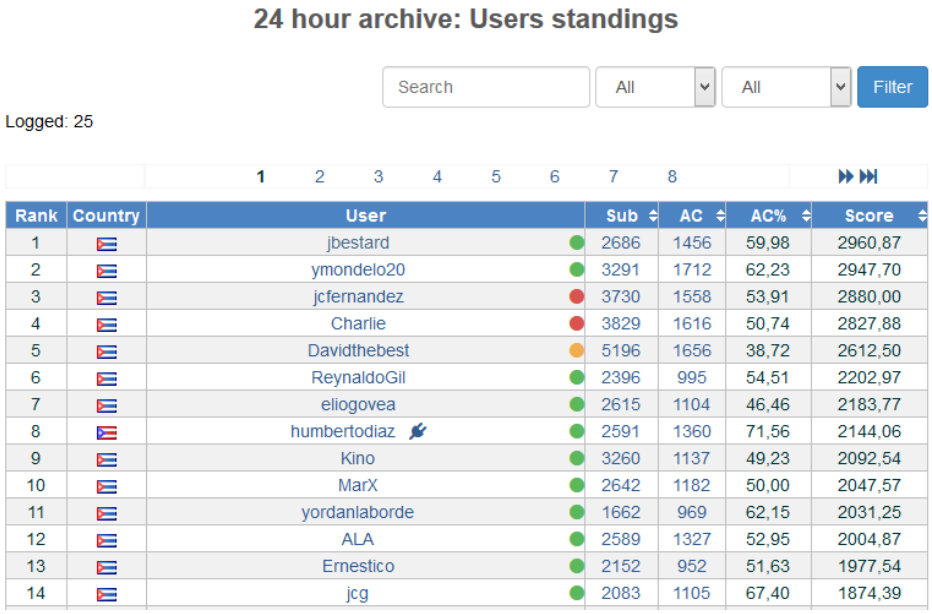
Figure 2. Typical interface to upload a problem solution in an online judge.

One of the most important advantages of POJs is that they allow users to select and try to solve the problems they consider suitable for improving their knowledge acquisition at their personalized pace. According to the amount of problems finally solved by each user, the POJ also publishes a ranking (Figure 3). The reaching of a top position in this ranking is an extra motivation for the users, being then this gamification environment a very good scenario for boosting the users' programming abilities and knowledge [3].

The wide acceptance of POJs and the increasing number of users that are currently solving a very high percentage of all the available problems, have implied an increment of such available problems. Therefore, it is currently difficult for several users to find out the appropriate problem for trying to solve, according to their current experience and learning needs. This is a typical information overloading scenario, in which users can experiment a high discouragement and frustration when the actual difficult level of the problems they are trying to solve does not match with their current knowledge profile and therefore it is difficult to develop a successful solution.

Nowadays, the classical solution to face these information overload problems is the development of a recommender system focused on the corresponding scenario. Recommender Systems goal is to provide users the items that best fit their preferences and needs in an overload search space [4–6]. They have been successfully used in scenarios such as tourism [7], e-commerce, e-learning, or social networks [8]. In the specific case of e-learning, there are also specific application scenarios such as

the recommendation in learning objects repositories [9], in learning management systems [10], or in the students' course recommendation [11]. However, in contrast to its wide use in other e-learning context, there have been few research efforts focused on the development of recommendation tools for POJs [12,13].



Rank	Country	User	Sub	AC	AC%	Score
1		jbestard	2686	1456	59,98	2960,87
2		ymondelo20	3291	1712	62,23	2947,70
3		jcfernandez	3730	1558	53,91	2880,00
4		Charlie	3829	1616	50,74	2827,88
5		Davidthebest	5196	1656	38,72	2612,50
6		ReynaldoGil	2396	995	54,51	2202,97
7		eliogovea	2615	1104	46,46	2183,77
8		humbertodiaz	2591	1360	71,56	2144,06
9		Kino	3260	1137	49,23	2092,54
10		MarX	2642	1182	50,00	2047,57
11		yordanlaborde	1662	969	62,15	2031,25
12		ALA	2589	1327	52,95	2004,87
13		Ernestico	2152	952	51,63	1977,54
14		jcg	2083	1105	67,40	1874,39

Figure 3. Typical interface showing the users' ranking in an online judge.

Regarding the e-learning systems research area, it is important to remark that the problem we are facing in the current research is different from the task related to users' guiding through the learning context, which is usually handled with very traditional approaches such as Intelligent Tutoring Systems or ontologies-supported approaches [14,15]. Generally, such approaches require as input an important amount of structured information characterizing e-learning content and user activity associated to the corresponding e-learning scenario, for guaranteeing an appropriate user characterization. At this moment such information is not available for typical POJs scenarios, making very difficult the use of traditional approaches such as the previous ones. Therefore, recommender systems will be used in this context as an appropriate solution for reducing information overload in these scenarios. Specifically, we will focus on collaborative filtering recommendation [16], which is a popular paradigm that does not depend on the availability of structured information.

This paper is then focused on the development of an uncertainty-aware framework for recommending problems to be solved in POJs, supporting and guiding the users in the search of problems they should be able to solve during their knowledge acquisition, and avoiding in this way failures and frustration. The management of the uncertainty associated to the user's behavior in the current scenario, would improve the recommendation generation process in relation to previous proposals that perform a crisp management of the user profile.

The main contributions of the paper are:

- A fuzzy based uncertainty-aware approach for recommending problems to solve in POJs.
- A data preprocessing strategy to be used on the user profile to remove anomalous behaviors that could affect the later recommendation generation.

This paper is structured as follows. Section 2 shows the needed background for the proposal presentation, including programming online judges, fuzzy tools in recommender systems, and recommender systems approaches for POJs. Section 3 presents the new uncertainty-aware proposal for problems recommendation in POJs, which also includes a novel data preprocessing

approach to be used in this scenario. Section 4 presents the evaluation of the proposal, including comparison against previous related works. Section 5 concludes the current research, pointing out future works.

2. Background

This section reviews in short different concepts about programming online judges, fuzzy tools in recommender systems, and the application of recommender systems in POJ scenarios. These research topics are of particular interest regarding the current contribution.

2.1. Programming Online Judges

POJs are typically web applications that contain a large list of programming problems to be solved. Several authors have reported their experiences with online judges for supporting programming subjects. As far as we know, the first references to POJs in literature were performed by Kurnia et al. [1], presenting POJs as useful tools for evaluating source codes that solve the proposed problems. Furthermore, Leal and Silva [17] developed more portable online judges applications that can be also used on-the-fly in programming contests. More recently, Llana et al. [18] introduced new administration facilities through new POJs interfaces; and Wang et al. [2] exposed their experiences on the integration of a POJ in a practice-oriented programming course. The literature has also reported the integration of POJs with learning management systems such as Moodle [19]. Beyond these representative works, Ala-Mutka [20] and Caiza and Del Amo [21] have developed more exhaustive surveys on tools for automatic evaluation of solutions to programming problems.

The current contribution presents a fuzzy-based approach for suggesting problems to be solved by the users in this scenario, assuming that the objective of the users in this kind of applications is to solve as many problems as they can, to improve the knowledge acquisition process. These users' goal in POJ has been previously documented by several authors [13,22,23], pointing out that this success implies a stronger degree of achievement and satisfaction [23], increasing then the learners' efforts and performance in knowledge acquisition according to a basic pedagogical rule ("learners' effort will increase if they are more satisfied") [22].

2.2. Fuzzy Tools in Recommender Systems

Recommender systems are software tools focused on providing users the information that best fits their preferences and needs, in a search space overloaded with possible options. They can be classified in two big families: the content-based recommendation approaches, and the collaborative filtering-based recommendation approaches [8]. Content-based approaches are mainly focused on using item attributes for building items' and users' profiles, and at last generating the recommendations depending on the matching degree between the current user and all the available items. In contrast, collaborative filtering generates the recommendations by matching the current user profiles with the profile of other users, for generating the most appropriate item recommendations. This matching could be through direct similarity calculation (in the case of memory-based collaborative filtering), or through the learning of more complex predictive models (in model-based collaborative filtering) [16]. Specifically, memory-based collaborative filtering has been a popular approach regarding it reaches an appropriate balance between its simplicity, justifiability, efficiency, and stability [24]. Therefore, this approach will be used as base for the current proposal.

Furthermore, the previous two families of recommender systems approaches have been explored by using several tools taken from the soft computing fields in recommendation scenarios. Specifically, a recent survey has showed that fuzzy logic tools have been successfully used in the last few years [25], both in content-based and collaborative filtering. Specifically, such survey refers to several research works that have effectively incorporated fuzzy modelling in memory-based collaborative filtering.

Regarding the relatively large amount of recommendation approaches using fuzzy tools [25], here we will focus on such works that also incorporate memory-based collaborative filtering,

being this group directly related to the current proposal. Particularly, Al-Shamri and Al-Ashwal [26] present a fuzzy-weighted Pearson correlation coefficient for collaborative recommender systems, Zhang et al. [27] introduce a recommendation approach which combines fuzzy extensions of user-based and item-based collaborative filtering, and Cheng and Wang [28] present a fuzzy recommender system based on the integration of subjective preferences and objective information, where the preferences are presented through a fuzzy linguistic model. Following similar ideas, Menhaj and Jamalzehi [29] propose a proximity-based similarity measure containing a fuzzy inference system that depends on homophily correlation and influence correlation, and Son [30] proposes a novel hybrid user-based method that integrates fuzzy similarity degrees between users based on the demographic data.

On the other hand, the recent literature has reported some efforts toward the application of fuzzy tools in e-learning recommender systems [25]. Former works propose frameworks for a personalized learning recommender system, which aims to help students to find learning materials they would need to read [31]. More recently, other works have presented a fuzzy tree-structured learning activity model and a learner profile model to comprehensively describe the complex learning activities and learner profiles [32,33]. Eventually, Myszkorowski and Zakrzewska [34] use fuzzy logic for creating groups of students with similar needs enabling to differentiate appropriately their environment features.

The works referred in this section suggest that the incorporation of fuzzy modeling could lead to positive results in collaborative filtering-based, e-learning recommendation. The proposal developed in the current paper, aims at using fuzzy tools to improve recommendations in POJs scenario.

2.3. Recommender Systems in POJs Scenarios

The POJs recommendation scenario is different from the typical e-commerce collaborative filtering scenario, because the relation user-item is much more complex in the case of POJs. Basically, while in e-commerce scenarios this relationship tends to be only modeled through a simple rating, in POJs the user usually needs several solution attempts to successfully solved the associated item (in this case the problems). Therefore, it is necessary to propose new approaches focused in this specific scenario.

However, even though the use of recommender systems have been currently expanded to several domains, there have been too few research efforts focused on the development of problems recommendation tools for POJs.

An early reported research of a recommendation approach in POJs uses a basic user-based collaborative filtering that considers the problems solved/not solved by the users which are similar to the active one [12]. The effectiveness of this method was evaluated with data collected from a POJ, proving that the collaborative filtering principle can perform well in this specific scenario.

Caro and Jimenez [35] have also explored several similarity-based approaches for recommender systems in POJs, specifically considering user-based and item-based similarity schemes. This work uses a graph representation for modelling the relationship between users and problems, and incorporates concepts from social network analysis theory to develop new recommendation strategies. The evaluation of the proposal is performed through a dataset gathered by an online judge developed by the authors.

Recently, Yera and Martínez [13] have completed a further detailed representation of the user profiles, by taking into account in addition to solved/not solved problems, the amount of previous failed attempts which occur before the final effective or ineffective problem solution. The experimental study developed in this work, proves that the development of advanced user similarity functions that consider this additional information leads to an improvement in the recommendation performance. In addition, this proposal also includes a classification scheme for users and problems according to the efforts required by users when they solve problems, for using this information later in the development of a data preprocessing step to eliminate anomalous users' interactions that would affect the recommendation accuracy.

This recent research exposes an emerging but effective framework for suggesting problems to solve in POJs. However, as a major challenge it has to improve the reliability and flexibility of the user profiling, considering it is supported on crisp models. Therefore, the current proposal introduces a new method centered on a more flexible and reliable user profile management, through the use of fuzzy tools to support recommendation generation. Moreover, it is presented a new data preprocessing approach that couples well with the new fuzzy logic-supported proposal.

3. Problems Recommendation in Programming Online Judges Using Fuzzy Tools

In this section it is introduced the fuzzy-based recommendation framework for POJs. This framework is composed by four phases (Figure 4):

1. **Data preparation**, to put data into a suitable format for recommendation generation (Section 3.1),
2. The **data preprocessing strategy** to remove anomalous behaviors over the initial data (Section 3.2),
3. The **fuzzy transformation** of the POJ data, to provide flexibility to the representation of the interaction between any user and problem (Section 3.3), and
4. The users similarity calculation and final **problems recommendation** (Section 3.4).

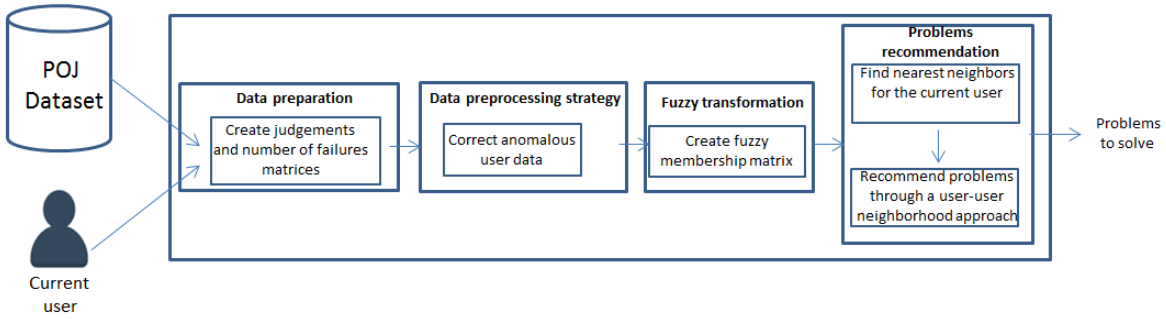


Figure 4. Framework for problems recommendation in POJ using fuzzy tools.

Table 1 summarizes the most relevant notation used in this section for our proposal.

Table 1. Main notation used across the proposal presentation.

Notation	Meaning
$M[u,p]$	Binary value indicating whether the user u solves problem p
$F[u,p]$	Number of attempted solutions the user u has tried over the problem p
$F^*[u,p]$	Fuzzy membership value associated to the number of failures $F[u,p]$
x_u	Average number of attempts for user u
dev_u	Average deviation of x_u
x_u^s	Average number of attempts for solved problems, by user u
dev_u^s	Average deviation of x_u^s
x_u^f	Average number of attempts for finally failed problems, by user u
dev_u^f	Average deviation of x_u^f
$C_{u,v}$	Set of problems which solutions have been attempted by both users u and v
$SimProb_p(u,v)$	Similarity between the users u and v according to a specific problem p
$Sim(u,v)$	Global similarity between two users u and v
w_p	Recommendation score for the problem p

3.1. Data Preparation

The current proposal receives as input the past performances of all the POJ users, i.e., a set of triplets $< u, p, j >$, each one representing an attempt of the user u for solving the problem p,

receiving as verdict the judgement j, which could be *accepted* if the problem was successfully solved, or *not accepted*, if not. The current work will not differentiate between the several kinds of failures mentioned in Introduction section (e.g., Wrong Answer, Time limited exceeded or Runtime Error).

In order to transform these data into a suitable format for recommendation generation in the POJ context, they are converted in two different matrices, M and F, with dimensions $n_u \times n_p$. Being n_u and n_p the number of users and problems. Such matrices respectively represent the judgements and the number of failures of each user, trying to solve each problem. Specifically, $M[u,p] = 1$ if the user u has successfully solved the problem p, and $M[u,p] = 0$ otherwise. In the case of F, $F[u,p]$ is the number of attempted solutions the user u has tried over the problem p, disregarding if it was finally solved or not.

3.2. Data Preprocessing Strategy

This section presents a simple but effective method for data preprocessing, to be applied on the F users' profiles data. Initially, it is introduced a *global* approach that considers the amount of solution attempts provided by a user u over a problem p. This approach is then extended into an *evaluation-based* approach that also takes into account whether the user u is successful in the resolution of the problem p or not.

A previous work referred in Section 2.3 has also focused on preprocessing POJs users' data [13]. However, such previously presented approach cannot be applied to the current proposal because it is based on crisp classifications of users trying to solve a specific problem (e.g., solved needing few attempts, not solved having too many attempts, and so on). In contrast, the aim of our current proposal is to avoid the use of crisp classifications, and then to perform an effective management of the uncertainty in the original POJ data through fuzzy tools. It is then necessary the use of further data preprocessing methods that could be coupled to the uncertainty-aware approach we are proposing here.

Therefore, our current work considers the outliers detection on the amount of users attempts trying to solve a specific problem, as the basic data preprocessing scheme to follow. Specifically, for each user, it is considered the problems where the number of attempts is out of the range $[\alpha \times (x_u - dev_u); \alpha \times (x_u + dev_u)]$, being x_u the average number of attempts for user u and dev_u the average deviation of such attempts. In addition, it is considered the coefficient $\alpha \approx 1$ for adding some flexibility to the definition of the range.

For the problems which are out of this range, we propose to replace the associated amount of attempts, with the x_u value (Equation (2)). This transformation is fully-represented in Equation (4), on the left side of the arrow appears the two conditions (problem p attempted by u, and $F[u,p]$ not in the mentioned range), and on the right side appears the transformation to do. We name this strategy as the *global* data preprocessing approach:

$$Attempted_u = \{set\ of\ problems\ p\ attempted\ by\ user\ u,\ i.e.,\ verifying\ F[u,p] > 0\} \tag{1}$$

$$x_u = \frac{\sum_{p \in Attempted_u} F[u,p]}{|Attempted_u|} \tag{2}$$

$$dev_u = \frac{\sum_{p \in Attempted_u} abs(F[u,p] - x_u)}{|Attempted_u|} \tag{3}$$

$$p \in Attempted_u\ and\ F[u,p] \notin [\alpha * (x_u - dev_u); \alpha * (x_u + dev_u)] \quad \rightarrow \quad F[u,p] = x_u \tag{4}$$

Alternatively, we also propose an *evaluation-based* data preprocessing strategy, in which the outliers detection is done independently first for the set of accepted (solved) problems, and then for the problems attempted but at last not solved. Here, two pairs are defined for each user:

1. A pair $[\alpha * (x_u^s - dev_u^s); \alpha * (x_u^s + dev_u^s)]$ characterizing accepted problems, and
2. A pair $[\alpha * (x_u^f - dev_u^f); \alpha * (x_u^f + dev_u^f)]$ characterizing problems attempted but failed.

Here $\alpha \approx 1$ is also used with the same purpose in relation to the previous strategy:

$$Solved_u = \{problems\ p\ solved\ by\ user\ u,\ i.e., verifying\ M[u,p] = 1\} \tag{5}$$

$$Failed_u = \{problems\ p\ failed\ by\ user\ u,\ i.e., verifying\ M[u,p] = 0\ and\ F[u,p] > 0\} \tag{6}$$

$$x_u^s = \frac{\sum_{p \in Solved_u} F[u,p]}{|Solved_u|} \tag{7}$$

$$dev_u^s = \frac{\sum_{p \in Solved_u} abs(F[u,p] - x_u^s)}{|Solved_u|} \tag{8}$$

$$x_u^f = \frac{\sum_{p \in Failed_u} F[u,p]}{|Failed_u|} \tag{9}$$

$$dev_u^f = \frac{\sum_{p \in Failed_u} abs(F[u,p] - x_u^f)}{|Failed_u|} \tag{10}$$

In both cases, if the amount of attempts associated to an accepted problem p falls out of the range $[\alpha * (x_u^s - dev_u^s); \alpha * (x_u^s + dev_u^s)]$, for recommendation purposes its amount of previous failures is replaced by x_u^s (Equation 11). On the other hand, if the problem was attempted but failed, and its amount of failures is out of the range $[\alpha * (x_u^f - dev_u^f); \alpha * (x_u^f + dev_u^f)]$, then it is replaced by x_u^f, for recommendation purposes (Equation (12)), being both conditions checked for all the users in the POJ dataset. Here x_u^s, dev_u^s, x_u^f, and dev_u^f will be always available for any user that has submitted solutions to the system, considering that they are calculated through $F[u,p]$ values (e.g., the current amount of solution attempts of user u over problem p), which is always an integer value.

$$p \in Solved_u\ and\ F[u,p] \notin [\alpha * (x_u^s - dev_u^s); \alpha * (x_u^s + dev_u^s)] \quad \rightarrow \quad F[u,p] = x_u^s \tag{11}$$

$$p \in Failed_u\ and\ F[u,p] \notin [\alpha * (x_u^f - dev_u^f); \alpha * (x_u^f + dev_u^f)] \quad \rightarrow \quad F[u,p] = x_u^f \tag{12}$$

Finally, we will consider a third strategy that performs the data preprocessing only over the problems failed by the user. Therefore, it only applies the Equation (12), disregarding the cases in which Equation (11) is verified. This strategy is named in the rest of the paper as the *failed-evaluation-based* strategy.

Remark 1. *We also evaluated a strategy that applies only Equation (11), disregarding the cases where Equation (12) is verified. However, it leads to poor results and therefore it was not included in the current paper.*

3.3. Fuzzy Transformation

In this section is introduced a fuzzy transformation whose input is the matrix F preprocessed in the previous step, such a transformation manages in a proper and flexible way the inherent vagueness and uncertainty associated to the users' attempts behaviour and information. This fuzzy modelling allows, as a concrete benefit to the current framework, a more robust representation of the users' behavior. Such robustness and flexibility cannot be easily handled through other comparable soft computing approaches [25], such as Bayesian networks, Markov models, or neural networks.

To provide flexibility to the modelling of the relationship between any user u and problem p, it is then considered the fuzzy set *difficult* that represents the difficulty level of a problem, which is represented through the membership function presented in Figure 5. As it is typically considered in this kind of fuzzy representation, it is very difficult the predefined initialization for the parameter t in the mentioned figure. This parameter is focused on characterizing the uncertainty level associated to each possible number of failures associated to some user in relation to a specific problem. Its optimal values will be analyzed as part of the experimental section.

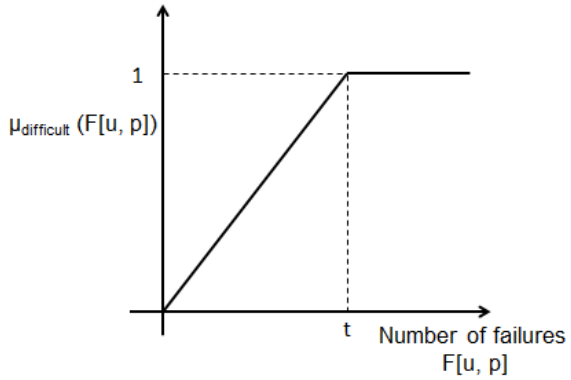


Figure 5. Membership function for the fuzzy set *difficult*.

Regarding this fuzzy set, the matrix F containing the number of failures for each user is then transformed into a matrix F^* which has the membership values associated to such kind of failures according to the fuzzy set difficult (Equation (13)).

$$F^*[u,p] = \mu_{difficult}(F[u,p]) \tag{13}$$

This new matrix F^* is actively used at the users similarity calculation in the next phase of our contribution.

This fuzzy representation through F^* matrix assumes that all problems with more than t attempts are considered of similar difficulty for the current user (i.e., the maximal difficulty). We want to remark that this assumption could not be considered through the former matrix F, that only represents a numeric scale which lacks of flexibility. We have performed additional experiments, not included in this paper, verifying that the execution of the procedure considering F^* outperforms the same procedure but considering F.

3.4. Users Similarity Calculation and Problems Recommendation

As a collaborative filtering recommendation approach, the current proposal needs to define a user similarity calculation scheme for considering the information available in the current scenario, in this case the M and F^* matrices. In order to obtain such scheme, first it is necessary to consider the set $C_{u,v}$ of problems which solutions have been attempted (successfully or unsuccessfully), by each pair of users u and v (Equation (14)).

$$C_{u,v} = \{set\ of\ problems\ where\ F^*[u,p] > 0\ and\ F^*[v,p] > 0\} \tag{14}$$

Afterwards, it is defined the similarity $SimProb_p(u,v)$ between the users u and v according to a specific problem p, taking as basis the absolute difference between the membership values in the F^* matrix, for both users at the problem p. Specifically, it will be considered for those problems located in the set $C_{u,v}$, and which have the same final verdict for both users ($(M[u,p] = M[v,p])$). In other cases, the value of $SimProb_p(u,v)$ will be zero. Equation (15) formalizes this similarity function.

$$SimProb_p(u,v) = \begin{cases} 1 - abs(F^*[u,p] - F^*[v,p]), if\ M[u,p] = M[v,p] \\ \qquad\qquad and \text{ problem } p \text{ is in } C_{u,v} \\ 0, \text{otherwise} \end{cases} \tag{15}$$

Finally, the global similarity between two users u and v is calculated as the average value for all the $SimProb_p(u,v)$ values, considering all the problems p in the set $C_{u,v}$ (Equation (16)).

$$Sim(u,v) = \frac{\sum_{p \in C_{u,v}} SimProb_p(u,v)}{|C_{u,v}|} \quad (16)$$

This similarity function is used by the current proposal, for finding the top k nearest neighbors for a current user u, to suggest him/her problems to solve.

These top k users are employed to calculate a score w_{up} for each problem p not solved by the active user u, as the sum of the similarities between u and the neighbors that solved p (Equation (17)). At last, the top n problems p with the highest scores w_{up} are recommended to u as the final recommendation list. This list can be composed by both problems previously attempted or not attempted by the user u, it will depend on problems attempted by the top k nearest neighbors.

$$w_{up} = \sum_{v \in top\_k(u)} Sim(u,v) \quad (17)$$

4. Experimental Study

The evaluation of the current proposal is done through a dataset obtained from the Caribbean Online Judge, composed of 1910 users, 584 problems, and around 148,000 user attempts. This dataset has been already used in previous research works presented in the background section [13].

Considering this dataset, the training and test sets were created according to a typical splitting procedure for this kind of recommender systems scenarios [36]. Specifically, it is based on a popular hold out scheme where all the solved problems for each user are randomly splitted in two sets which are respectively appended to the training set and the test set. As a feature of the hold out protocol, this splitting procedure was developed 10 times, composing 10 pairs of training-test sets. The recommendation accuracy of each set is calculated independently, and the average of the accuracy values are then showed in the results presented in the current section.

The F1 measure will be used for evaluating the generated recommendations, being a popular evaluation criteria for the top-n recommendation generation task [36]. F1 (Equation (18)) is defined in terms of Precision and Recall measures (Equations (19) and (20)), that in addition, as it is presented in Table 2, depend on the amount of recommended problems that were solved (precision) and the amount of solved problems that were recommended (recall).

$$F1 = \frac{2 * precision * recall}{precision + recall} \quad (18)$$

$$precision = \frac{\#tp}{\#tp + \#fp} \quad (19)$$

$$recall = \frac{\#tp}{\#tp + \#fn} \quad (20)$$

Table 2. Possible recommendation results, regarding solved and recommended problems.

	Recommended	**Not Recommended**
Solved	True-Positive (tp)	False-Negative (fn)
Not solved	False-Positive (fp)	True-Negative (tn)

Specifically, the evaluation protocol is performed by obtaining a list of recommended problems for each user, considering training set data. Such list, as well as the solved problems at the test set, are used for calculating F1 for each user. Finally, the global F1 value is calculated by averaging the F1 values obtained for each user.

The current proposal depends on three main parameters:

- The amount of k nearest neighbors: It will be used $k = 80$, considering it is a value previously used in related works presented in the background section. As future works, it will be explored the behavior of the proposal for other values of k.
- The value of t, in the fuzzy membership function presented in Figure 5. This value is connected to the number of attempts where the difficulty of the problems can be considered as maximum (see Section 3.3). Therefore, it could be linked with the global average of x_u, being x_u the average number of attempts for user u (see Table 1). Regarding that the average of x_u for the current data was 1.18 with a mean deviation of 1.17, here it will be considered $t \geq 3$. Specifically, it will be reported the results for $t = 4$, $t = 5$, and $t = 6$. Further experiments were developed considering other values (e.g., $t = 3$), but their performances were under the associated to the mentioned ones.
- The value of α in the data preprocessing strategy. To verify whether the flexibility in the definition of the interval for the outliers characterization leads to positive results, it will be considered $\alpha = 1$ (no flexibility), and $\alpha = 0.8$ (a small flexibility, considering correct values to some small amounts of attempts treated as outliers by $\alpha = 1$).

The current experiments have two main objectives. In both cases it will be considered two scenarios regarding the size of the recommendation list:

1. At first it will be considered recommendation lists with large sizes, varying such size n in the range [5, 25] with step 10, and
2. At second it will be considered smaller recommendation lists, varying n in the range [1, 5] with step 1.

Objective 1: Determine whether the fuzzy recommendation approach outperforms previous approaches that do not consider the management of the uncertainty associated to POJ data.

In order to accomplish this objective, we compare the accuracy (F1) of the current proposal (without the data preprocessing step), in relation to the proposals presented by Yera and Caballero [12] (*UCF-OJ* in the results tables), and Yera and Martínez [13], in the last case also without considering the data preprocessing step (*UCF-OJ++* in the tables). It is also included a comparison against a successful approach presented at [35], specifically the problem-based approach using as similarity the weight of the relation between two problems, regarding the social network analysis-related strategies presented in the mentioned [35] (*PCF-OJ* in the tables). Other approaches in [35] perform worse in this scenario and then were not included here. As it was explained before, the current proposal considers $t = 4$, $t = 5$, and $t = 6$ as possible values of the parameter t, being identified as *Fuzzy-UCF-OJ* in the mentioned tables.

Tables 3 and 4 show the results of this comparison for larger and smaller recommendation lists, respectively. As it was expected the performance associated to the new proposal is similar or better than the related to the previous approaches. Specifically, the more notable improvements were obtained for $n = 5$ in larger recommendation lists, and for all cases in smaller recommendation lists. However, for larger values of n the improvement becomes modest in relation to Yera and Martínez [13]. Furthermore, the best results of the proposal were obtained for $t = 6$ and $t = 5$, leaving for $t = 4$ a lower accuracy value.

Table 3. Comparison between the current proposal and previous approaches, without considering data preprocessing step at Section 3.2. Large recommendation lists.

	5	**15**	**25**
UCF-OJ [12]	0.3568	0.3555	0.3237
PCF-OJ [35]	0.3187	0.2965	0.2657
UCF-OJ++ [13]	0.3613	0.3610	0.3277
Fuzzy-UCF-OJ (t = 6)	**0.3663**	0.3600	0.3281
Fuzzy-UCF-OJ (t = 5)	0.3640	**0.3613**	**0.3286**
Fuzzy-UCF-OJ (t = 4)	0.3625	0.3607	0.3279

Table 4. Comparison between the current proposal and previous approaches, without considering data preprocessing step at Section 3.2. Smaller recommendation lists.

	1	**2**	**3**	**4**	**5**
UCF-OJ [12]	0.2454	0.2990	0.3316	0.3471	0.3568
PCF-OJ [35]	0.2327	0.2869	0.3127	0.3182	0.3187
UCF-OJ++ [13]	0.2407	0.2958	0.3268	0.3469	0.3613
Fuzzy-UCF-OJ (t = 6)	**0.2494**	0.3010	0.3305	**0.3517**	**0.3663**
Fuzzy-UCF-OJ (t = 5)	0.2487	**0.3024**	**0.3320**	0.3512	0.3640
Fuzzy-UCF-OJ (t = 4)	0.2466	0.3021	0.3297	0.3488	0.3625

It is also remarkable the poor performance associated to the work [35]. In this case, this approach was originally evaluated in a different dataset in relation to the used in the current proposal. Therefore, we think that its positive performance previously reported by [35], could be closely related to the nature of data where it is used. Furthermore, this mentioned work disregards the number of previous attempts as a valuable information for recommendation generation, being proved later in [13] that this information could be important in such task. We think that this fact could justify this unexpected behavior, although a future analysis should be developed to support these assumptions and to fully exploit the results exposed by [35].

Table 5 presents the statistical comparison using Wilcoxon test [36], between the previous work that performs best in the state-of-art [13], and the three evaluated proposals (with $t = 4$, $t = 5$, and $t = 6$). For all cases, our proposals statistically outperform the previous work ($p < 0.05$).

Table 5. Statistical analysis of the results, without considering data preprocessing at Section 3.2. Wilcoxon test ($p < 0.05$).

	Significance Values
Fuzzy-UCF-OJ (t = 6) vs. UCF-OJ++ [13]	0.043
Fuzzy-UCF-OJ (t = 5) vs. UCF-OJ++ [13]	0.018
Fuzzy-UCF-OJ (t = 4) vs. UCF-OJ++ [13]	0.043

Objective 2: Determine the effect of the proposed data preprocessing strategy, in contrast to previous approaches that also consider data preprocessing in POJ scenarios.

To accomplish this objective it will be considered the use of the data preprocessing strategy previously presented in this work, to clean the original data and therefore improve the recommendation accuracy. With this aim in mind, it will be considered the values of t and α previously mentioned in this section, together with the three data preprocessing schemes explained in the previous section (global, evaluation-based, and failed-evaluation-based). In the failed-evaluation-based strategy, it was only considered $t = 6$ because other values obtained poor results and therefore they were not reported. We also consider a comparison against a previous related work that also considers data preprocessing [13], which is identified as *UCF-OJ++ + preproc* in the tables. Our current proposal in this case is identified as *Fuzzy-UCF-OJ + preproc*.

The analysis of the presented results for large recommendation lists (Table 6), at first concludes that for smaller sizes of such list the proposal introduces an improvement in the recommendation accuracy (e.g., for $n = 5$, the best F1 value for the proposal was 0.3674, while for the previous work was 0.3615). However, for larger sizes the improvements become more modest.

Additionally, it is worthy to note that although for higher values of n the best performance was obtained for the global strategy for $t = 6$ and $\alpha = 0.8$; for smaller values in such Table 6 (where the improvement is more relevant), it is not a specific execution scenario that leads to the achievement of the best F1 values. Here, the best performance was obtained for the failed-evaluation-based strategy with $t = 6$ and $\alpha = 1$ (in $n = 5$), and for the evaluation-based strategy also with $t = 6$ and $\alpha = 1$ (in $n = 15$).

Globally, as an unexpected finding in large recommendation lists, it was detected that the most sophisticated data preprocessing strategy (the evaluation-based), does not necessarily lead to the reaching of a better recommendation accuracy.

Table 6. Comparison between the current proposal and previous approaches, considering data preprocessing. Large recommendation lists.

	5	**15**	**25**
UCF-OJ++ + preproc [13]	0.3615	0.3619	0.3280
Fuzzy-UCF-OJ + preproc(t = 6, α = 1, global)	0.3652	0.3607	0.3288
Fuzzy-UCF-OJ + preproc (t = 5, α = 1, global)	0.3652	0.3604	0.3285
Fuzzy-UCF-OJ + preproc (t = 4, α = 1, global)	0.3643	0.3612	0.3288
Fuzzy-UCF-OJ + preproc (t = 6, α = 0.8, global)	0.3649	0.3624	**0.3291**
Fuzzy-UCF-OJ + preproc (t = 5, α = 0.8, global)	0.3649	0.3614	0.3289
Fuzzy-UCF-OJ + preproc (t = 4, α = 0.8, global)	0.3640	0.3606	0.3288
Fuzzy-UCF-OJ + preproc (t = 6, α = 1, eval-based)	0.3633	**0.3626**	0.3280
Fuzzy-UCF-OJ + preproc (t = 5, α = 1, eval-based)	0.3649	0.3609	0.3273
Fuzzy-UCF-OJ + preproc (t = 4, α = 1, eval-based)	0.3662	0.3611	0.3275
Fuzzy-UCF-OJ + preproc (t = 6, α = 0.8, eval-based)	0.3659	0.3620	0.3279
Fuzzy-UCF-OJ + preproc (t = 5, α = 0.8, eval-based)	0.3644	0.3601	0.3274
Fuzzy-UCF-OJ + preproc (t = 4, α = 0.8, eval-based)	0.3667	0.3605	0.3268
Fuzzy-UCF-OJ + preproc (t = 6, α = 1, failed-eval-based)	**0.3674**	0.3620	0.3290
Fuzzy-UCF-OJ + preproc (t = 6, α = 0.8, failed-eval-based)	0.3664	0.3613	0.3287

In contrast to the results obtained for large recommendation lists, in the case of smaller recommendation lists with sizes lying in the range [1, 5] (Table 7), for all cases it was obtained that the more sophisticated data preprocessing strategies lead to the reaching of the best results (i.e., evaluation-based and failed-evaluation-based).

Table 8 presents the statistical comparison using Wilcoxon test [36], between the previous work that performs best in the state-of-art [13], and four evaluated proposals which have positive performance across all the parameter settings. For all cases, our proposals statistically outperforms the previous work ($p < 0.05$).

Table 7. Comparison between the current proposal and previous approaches, without considering data preprocessing. Smaller recommendation lists.

	1	**2**	**3**	**4**	5
UCF-OJ++ + preproc [13]	0.2430	0.2967	0.3268	0.3485	0.3615
Fuzzy-UCF-OJ + preproc (α = 1, t = 6, global)	0.2501	0.3044	0.3302	0.3523	0.3652
Fuzzy-UCF-OJ + preproc (α = 1, t = 5, global)	0.2481	0.3038	0.3313	0.3515	0.3652
Fuzzy-UCF-OJ + preproc (α = 1, t = 4, global)	0.2471	0.3017	0.3318	0.3505	0.3643
Fuzzy-UCF-OJ + preproc (α = 0.8, t = 6, global)	0.2479	0.3029	0.3299	0.3532	0.3649
Fuzzy-UCF-OJ + preproc (α = 0.8, t = 5, global)	0.2493	0.3026	0.3318	0.3525	0.3649
Fuzzy-UCF-OJ + preproc (α = 0.8, t = 4, global)	0.2476	0.3014	0.3308	0.3511	0.3640
Fuzzy-UCF-OJ + preproc (α = 1, t = 6, eval-based)	**0.2545**	0.3047	0.3296	0.3529	0.3633
Fuzzy-UCF-OJ + preproc (α = 1, t = 5, eval-based)	0.2509	0.3051	0.3307	0.3520	0.3649
Fuzzy-UCF-OJ + preproc (α = 1, t = 4, eval-based)	0.2535	0.3035	0.3316	0.3513	0.3662
Fuzzy-UCF-OJ + preproc (α = 0.8, t = 6, eval-based)	0.2518	0.3020	0.3326	0.3541	0.3659
Fuzzy-UCF-OJ + preproc (α = 0.8, t = 5, eval-based)	0.2530	0.3058	0.3318	0.3538	0.3644
Fuzzy-UCF-OJ + preproc (α = 0.8, t = 4, eval-based)	0.2521	**0.3059**	0.3304	0.3520	0.3667
Fuzzy-UCF-OJ + preproc (α = 1, t = 6, failed-eval-based)	0.2530	0.3050	0.3334	**0.3549**	**0.3674**
Fuzzy-UCF-OJ + preproc (α=0.8, t = 6, failed-eval-based)	0.2510	0.3045	**0.3365**	0.3540	0.3664

Table 8. Statistical analysis of the results, considering data preprocessing at Section 3.2. Wilcoxon test ($p < 0.05$).

	Significance Values
Fuzzy-UCF-OJ + preproc ($t = 6$, $\alpha = 0.8$, global) vs. UCF-OJ++ + preproc [13]	0.018
Fuzzy-UCF-OJ + preproc ($t = 6$, $\alpha = 1$, eval-based) vs. UCF-OJ++ + preproc [13]	0.028
Fuzzy-UCF-OJ + preproc ($t = 6$, $\alpha = 1$, failed-eval-based) vs. UCF-OJ++ + preproc [13]	0.018
Fuzzy-UCF-OJ + preproc ($t = 6$, $\alpha = 0.8$, failed-eval-based) vs. UCF-OJ++ + preproc [13]	0.028

Overall, these last results suggest that the use of techniques for preprocessing the data in the current POJ scenario can improve the recommendation accuracy, in a similar way to other preprocessing approaches for recommendation scenarios [37,38].

Beyond the two main objectives of the current evaluation, mentioned at the beginning of this section and focused on the accuracy of the proposal, we also focus briefly on analyzing the amount of data modified by the data preprocessing strategies. Table 9 presents the percentage of data modified in the data preprocessing step. In the case of the global and the failed-eval-based strategies the values were always under 17%, which guarantee a low intrusiveness level and match with values previously obtained in similar works according to this criteria [39]. For the eval-based strategy, the intrusion degree was higher.

Table 9. Percentage of data modified during the preprocessing step.

Fuzzy-UCF-OJ + Preproc (Global)	
$\alpha = 0.8$	**$\alpha = 1$**
14.82%	8.74%
Fuzzy-UCF-OJ + preproc (eval-based)	
$\alpha = 0.8$	**$\alpha = 1$**
36.69%	28.3%
Fuzzy-UCF-OJ + preproc (failed-eval-based)	
$\alpha = 0.8$	**$\alpha = 1$**
16.09%	13.44%

Summarizing, the globally-obtained results suggest that the current proposal can be associated to a POJ for generating effective problems' recommendations. Furthermore, it is important to point out that such proposal could be coupled with other very popular e-learning frameworks, being Massive Open Online Courses (MOOCs) a key example of such platforms. MOOCs and POJs have as common features, the fact that both contain e-learning content where users have to access at their personalized pace. However, while current POJs are mainly focused on programming practices, the users at MOOCs are usually learning and practising simultaneously inside the platform. Therefore, it is expected larger levels of participation in MOOCs, in relation to POJs. This increasing number of users leads to a sparser user preference dataset, which could make necessary the transformation of the current proposal, to be used in this new scenario. In fact, recent works have faced the recommendation in MOOCs through different innovative approaches such as the reciprocal recommendation [40], or the use of information from external sources [41]. Our future work will then explore these approaches to extend the current proposal to guarantee a successful tentative integration with MOOC-like frameworks. Furthermore, with the same aim in mind, it will be developed user studies for a better understanding of their motivations and needs.

In other direction, future works will also consider an adaptive modification of the size n of the top n recommendation list, varying it according to past user behavior. This approach has been successfully used in previous works such as [42].

5. Conclusions

The current contribution has been focused on the incorporation of fuzzy logic tools for proposing an approach to recommend problems to solve in POJs scenarios. This approach also incorporates a data preprocessing strategy to correct anomalous users' behavior which could affect the recommendation generation. A relevant feature of the current work is the management of the uncertainty associated to the POJ scenario, not performed previously as far as we know. The development of the experimental evaluation of the proposal leads to the following conclusions:

- Both the fuzzy approach and the data preprocessing strategy outperform previous related works in relation to the accuracy of the problem recommendation task, according to the F1 metric.
- The best performances were globally associated to smaller sizes of recommendation list ($n \leq 5$).
- The use of more sophisticated data preprocessing strategy (the evaluation-based strategy), leads to the best results in small sizes of recommendation list ($n \leq 5$).
- The global and the failed-eval-based data preprocessing strategies modify always under 17% of the data, guaranteeing a low intrusiveness level of the proposal.

Next future work will explore: (1) the adaptive modification of the size n of the top n recommendation list, (2) the integration of the current framework in MOOCs scenarios, as well as (3) considering the behavior of the users across the time. Furthermore, we will explore the use of alternative recommendation techniques in the current scenario, that would allow a balance between recommendation accuracy, efficiency, diversity, and transparency.

Acknowledgments: This research work was partially supported by the Research Project TIN2015-66524-P, and a Scholarship for Young Doctors (2017) provided by CEATIC, UJaen, Spain.

Author Contributions: The three authors contributed to the design of the proposal and the experiments; R.Y.T. implemented the proposal and run the experiments. Eventually, the three authors contributed to the analysis of the results and to the conclusions from them.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kurnia, A.; Lim, A.; Cheang, B. Online judge. *Comput. Educ.* **2001**, *36*, 299–315.
2. Wang, G.P.; Chen, S.Y.; Yang, X.; Feng, R. OJPOT: Online judge & practice oriented teaching idea in programming courses. *Eur. J. Eng. Educ.* **2016**, *41*, 304–319.
3. Seaborn, K.; Fels, D.I. Gamification in theory and action: A survey. *Int. J. Hum.-Comput. Stud.* **2015**, *74*, 14–31.
4. Bobadilla, J.; Ortega, F.; Hernando, A.; Gutiérrez, A. Recommender systems survey. *Knowl.-Based Syst.* **2013**, *46*, 109–132.
5. Ortega, F.; Hernando, A.; Bobadilla, J.; Kang, J.H. Recommending items to group of users using Matrix Factorization based Collaborative Filtering. *Inf. Sci.* **2016**, *345*, 313–324.
6. Venkatraman, S. A Proposed Business Intelligent Framework for Recommender Systems. *Informatics* **2017**, *4*, 40.
7. Noguera, J.; Barranco, M.; Segura, R.; Martínez, L. A Mobile 3D-GIS Hybrid Recommender System for Tourism. *Inf. Sci.* **2012**, *215*, 37–52.
8. Lu, J.; Wu, D.; Mao, M.; Wang, W.; Zhang, G. Recommender system application developments: A survey. *Decis. Support Syst.* **2015**, *74*, 12–32.
9. Sergis, S.; Sampson, D. Learning Object Recommendations for Teachers Based On Elicited ICT Competence Profiles. *IEEE Trans. Learn. Technol.* **2016**, *9*, 67–80.
10. Salehi, M.; Kamalabadi, I. Hybrid recommendation approach for learning material based on sequential pattern of the accessed material and the learner's preference tree. *Knowl.-Based Syst.* **2013**, *48*, 57–69.
11. Sobecki, J. Comparison of Selected Swarm Intelligence Algorithms in Student Courses Recommendation Application. *Int. J. Softw. Eng. Knowl. Eng.* **2014**, *24*, 91–109.
12. Yera, R.; Caballero Mota, Y. An e-Learning Collaborative Filtering Approach to Suggest Problems to Solve in Programming Online Judges. *Int. J. Distance Educ. Technol.* **2014**, *12*, 51–65.
13. Yera, R.; Martínez, L. A recommendation approach for programming online judges supported by data preprocessing techniques. *Appl. Intell.* **2017**, *47*, 277–290.

14. Badaracco, M.; Martínez, L. A fuzzy linguistic algorithm for adaptive test in Intelligent Tutoring System based on competences. *Expert Syst. Appl.* **2013**, *40*, 3073–3086.
15. Miranda, S.; Orciuoli, F.; Sampson, D.G. A SKOS-based framework for Subject Ontologies to improve learning experiences. *Comput. Hum. Behav.* **2016**, *61*, 609–621.
16. Adomavicius, G.; Tuzhilin, A.T. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.* **2005**, *17*, 734–749.
17. Leal, J.P.; Silva, F. Mooshak: A Web-based multi-site programming contest system. *Softw. Pract. Exp.* **2003**, *33*, 567–581.
18. Llana, L.; Martin-Martin, E.; Pareja-Flores, C.; Velázquez-Iturbide, J.Á. FLOP: A User-Friendly System for Automated Program Assessment. *J. Univers. Comput. Sci.* **2014**, *20*, 1304–1326.
19. Verdú, E.; Regueras, L.M.; Verdú, M.J.; Leal, J.P.; de Castro, J.P.; Queirós, R. A distributed system for learning programming on-line. *Comput. Educ.* **2012**, *58*, 1–10.
20. Ala-Mutka, K.M. A survey of automated assessment approaches for programming assignments. *Comput. Sci. Educ.* **2005**, *15*, 83–102.
21. Caiza, J.; Del Amo, J. Programming Assignments Automatic Grading: Review of Tools and Implementations. In Proceedings of the INTED 2013, Valencia, Spain, 4–6 March 2013; pp. 5691–5700.
22. Nadolski, R.J.; Van den Berg, B.; Berlanga, A.J.; Drachsler, H.; Hummel, H.G.; Koper, R.; Sloep, P.B. Simulating light-weight personalised recommender systems in learning networks: A case for pedagogy-oriented and rating-based hybrid recommendation strategies. *J. Artif. Soc. Soc. Simul.* **2009**, *12*, 4.
23. Regueras, L.M.; Verdú, E.; Muñoz, M.F.; Pérez, M.A.; De Castro, J.P.; Verdú, M.J. Effects of competitive e-learning tools on higher education students: A case study. *IEEE Trans. Educ.* **2009**, *52*, 279–285.
24. Ning, X.; Desrosiers, C.; Karypis, G. A Comprehensive Survey of Neighborhood-Based Recommendation Methods. In *Recommender Systems Handbook*; Springer: Berlin, Germany, 2015; pp. 37–76.
25. Yera, R.; Martínez, L. Fuzzy Tools in Recommender Systems: A Survey. *Int. J. Comput. Intell. Syst.* **2017**, *10*, 776–803.
26. Al-Shamri, M.Y.H.; Al-Ashwal, N.H. Fuzzy-weighted Pearson Correlation Coefficient for Collaborative Recommender Systems. In Proceedings of the 15th International Conference on Enterprise Information Systems (ICEIS), Angers, France, 4–7 July 2013; pp. 409–414.
27. Zhang, Z.; Lin, H.; Liu, K.; Wu, D.; Zhang, G.; Lu, J. A hybrid fuzzy-based personalized recommender system for telecom products/services. *Inf. Sci.* **2013**, *235*, 117–129.
28. Cheng, L.C.; Wang, H.A. A fuzzy recommender system based on the integration of subjective preferences and objective information. *Appl. Soft Comput.* **2014**, *18*, 290–301.
29. Menhaj, M.B.; Jamalzehi, S. Scalable user similarity estimation based on fuzzy proximity for enhancing accuracy of collaborative filtering recommendation. In Proceedings of the 2016 4th International Conference on Control, Instrumentation, and Automation (ICCIA), Qazvin, Iran, 27–28 January 2016; pp. 220–225.
30. Son, L.H. HU-FCF: A hybrid user-based fuzzy collaborative filtering method in Recommender Systems. *Expert Syst. Appl.* **2014**, *41*, 6861–6870.
31. Lu, J. A Personalized e-Learning Material Recommender System. In Proceedings of the 2nd International Conference on Information Technology for Applications, Harbin, China, 9–11 January 2004; pp. 1–6.
32. Wu, D.; Lu, J.; Zhang, G. A Fuzzy Tree Matching-Based Personalized E-Learning Recommender System. *IEEE Trans. Fuzzy Syst.* **2015**, *23*, 2412–2426.
33. Poorni, G.; Balaji, K.; DeepthiNivetha, C. A Personalized E-Learning Recommender System Using the Concept of Fuzzy Tree Matching. *Int. J. Adv. Res. Comput. Eng. Technol.* **2015**, *4*, 4039–4043.
34. Myszkorowski, K.; Zakrzewska, D. Using Fuzzy Logic for Recommending Groups in E-Learning Systems. In *Computational Collective Intelligence. Technologies and Applications*; Springer: Berlin, Germany, 2013; pp. 671–680.
35. Caro, M.; Jimenez, G. Similar Users or Similar Items? Comparing Similarity-Based Approaches for Recommender Systems in Online Judges. In Proceedings of the ICCBR 2017, Rondheim, Norway, 26–28 June 2017; Springer: Berlin, Germany, 2017; pp. 92–107.
36. Gunawardana, A.; Shani, G. A Survey of Accuracy Evaluation Metrics of Recommendation Tasks. *J. Mach. Learn. Res.* **2009**, *10*, 2935–2962.
37. Castro, J.; Yera, R.; Martínez, L. An empirical study of natural noise management in group recommendation systems. *Decis. Support Syst.* **2017**, *94*, 1–11, doi:10.1016/j.dss.2016.09.020.

38. Castro, J.; Yera, R.; Martínez, L. A fuzzy approach for natural noise management in group recommender systems. *Expert Syst. Appl.* **2018**, *94*, 237–249.
39. Yera, R.; Caballero Mota, Y.; Martínez, L. Correcting noisy ratings in collaborative recommender systems. *Knowl.-Based Syst.* **2015**, *76*, 96–108.
40. Prabhakar, S.; Spanakis, G.; Zaïane, O. Reciprocal Recommender System for Learners in Massive Open Online Courses (MOOCs). In Proceedings of the International Conference on Web Learning (ICWL), Cape Town, South Africa, 20–22 September 2017; Springer: Berlin, Germany, 2017; pp. 157–167.
41. Symeonidis, P.; Malakoudis, D. MoocRec.com: Massive Open Online Courses Recommender System. In *RecSys Posters, Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, 15–19 September 2016*; ACM: New York, NY, USA, 2016.
42. De Meo, P.; Quattrone, G.; Terracina, G.; Ursino, D. An XML-Based Multiagent System for Supporting Online Recruitment Services. *IEEE Trans. Syst. Man Cybern. Part A* **2007**, *37*, 464–480.