Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 08, No. 02, June 2022.

RULE-BASED APPROACH FOR CONTEXT-AWARE COLLABORATIVE RECOMMENDER SYSTEM

Soulef Benhamdi[1], Abdesselam Babouri[2], Raja Chiky[3] and Jamal Nebhen[4]

(Received: 5-Jan.-2022, Revised: 5-Apr.-2022, Accepted: 22-Apr.-2022)

ABSTRACT

Sparsity is a serious problem of collaborative filtering (CF) that has a considerable effect on recommendation quality. Contextual information is introduced in traditional recommendation systems besides users' and items' information to overcome this problem. Several research works proved that incorporating contextual information may increase sparse data. For this, data-mining techniques are among the most effective solutions that have been used in context-aware recommendation systems to handle the sparsity problem. This paper proposes the combination of a new context-user-based similarity collaborative filtering recommendation technique with data mining techniques, as a solution to this problem and develops a novel recommendation system: Rule-based Context-aware Recommender System (R_CARS). R_CARS is experimented introducing four rule-based algorithms: JRip, PART, J48 and RandomForest, on four different datasets: DePaulMovie, InCarMusic, Restaurant and LDOS_CoMoDa and compared with the state-of-the-art models. The results of the experiment show that weighting the rating-based similarity with context and combining it with a rule-based technique can overcome the sparsity problem and significantly improve the accuracy of recommendation compared to the state-of-the-art models.

KEYWORDS

Sparsity, CARS, Rule-based recommendation systems, Data mining, Collaborative filtering, Similarity.

1. INTRODUCTION

Recommendation systems (RSs) are considered as the most effective solution to the problem of information overload in online systems [1]. RSs use information on users and items for providing recommendations to the target user. Three main approaches are used by this kind of system [2]-[3]:

- Content-based filtering generates recommendations to a user with regard to his/her preferences in the past.
- Collaborative filtering models are built from collected user-item interactions. By analyzing behavioural patterns across the whole user base, the recommendations are based on extracting interactions similarities among users, items or both.
- Hybrid recommender systems combine various methods, exploiting the benefits of each one.

Recent research indicated that users' preferences change according to the context [4]. For example, a user may choose a different kind of movie if he or she is going to watch the movie at *home* rather than at the *cinema*. Therefore, contextual factors are introduced to alleviate some traditional RS issues, especially sparsity [5] and consequently enhance the recommendation accuracy. In fact, incorporating contextual information may not alleviate the sparsity problem, but may reduce its effect by reducing sparse data, because " it is not guaranteed, under the same contextual situation, that users rate all the items" [6]. For this, several approaches are suggested to face sparsity problems in context-aware recommendation systems (CARSs): selection of influential contextual attributes [6], similarity-based collaborative filtering [7], data-mining techniques: sequencing patterns, classification, association, regression, Computational Intelligence (CI) techniques, …etc. [8]-[13].

There are three main techniques used to develop a context-aware recommendation system according to the step where the contextual information is integrated [14]: *pre-filtering, post-filtering and contextual modeling*. In the *pre-filtering* technique [15] (Figure 1 (a)), R represents users' ratings, C represents contextual information and contextual information is used as input data with ratings to generate recommendations; i.e., irrelevant ratings are filtered out using contexts. Afterward, the recommendation model is built using classic recommendation approaches. *Post-filtering* (Figure 1 (b)) [16], one of the

1. S. Benhamdi is with Dept. of Computer Sci., Uni. 8 Mai 1945, Guelma, Algeria. Email: benhamdi.soulef@univ-guelma.dz
2. A. Babouri is with LGEG Laboratory, Uni. 8 Mai 1945, Guelma, Algeria. Email: babouri.abdesselam@univ-guelma.dz
3. R. Chiky is with ISEP, Paris, France. Email: raja.chiky@isep.fr
4. J. Nebhen is with College of Computer Sci. and Eng., Prince Sattam bin Abdulaziz Uni., KSA. Email: j.nebhen@psau.edu.sa

classic recommendation models, is used to predict ratings. Afterward, contextual data is used to adjust the predicted ratings. *Contextual modeling* technique (Figure 1 (c)) [17] incorporates contextual information in the ranking function to calculate predictions.

The contributions of this paper are to;
- Design a pre-filtering context-aware recommender system framework: Rule-based Context-aware Recommender System (R_CARS), that is based on context-user-based similarity collaborative filtering and a rule-based technique. The rating-based similarity is weighted using context-based similarity.
- Alleviate data sparsity using a rule-based approach with a context-based similarity technique.
- For comparative raisons, an experiment is conducted with the most popular classifiers: ZeroR, OneR, REPTree, RandomTree, JRip, PART, J48 and RandomForest. The four last classifiers are selected based on their performance in term of accuracy to be presented in this paper. The proposed model is compared with the state-of-the-art models.

The first section gives an overview of recommendation systems and research aims. Section two presents related work, whereas section three details the proposed approach. Section four presents the experiment results. Section five concludes this paper.

2. RELATED WORK

Context-awareness in recommendation systems is a novel research domain that attracted researchers in the last decade [18]-[19]. In real life, the users' choices may vary from one situation to another; for example, time, companion and location are important factors that users may take into account when selecting a movie [20]. The user's activity (e.g. driving) or emotional attributes (e.g. mood) may affect his/her choice when listening to music [21]. Location, time, type of food served, price of the meal…, might be important factors that should be involved in recommending services (hotels, restaurants, …etc.) to tourists [22].

This paper focuses on research works that exploit context-similarity and/or rule-based approaches to predict ratings. To face the sparsity and cold start problems, a recommendation algorithm (called TCAR) that uses association rule mining and an improved overlapping community detection method is proposed in [23]. Time-weighted similarity between users is computed to detect overlapping communities. Association rule mining is employed to model users' drifts over time. However, this method takes into consideration only a single context attribute. [24] suggests a rule-based recommendation method to predict ratings. To reduce sparse data, this method aims to reduce the number of context features; i.e., construct low-dimensional latent contexts. [25] proposes a context similarity measure within a multi-criteria collaborative recommendation approach for service recommendation. To prevent sparsity, ratings are used within similar contexts. A ranking-based recommendation algorithm (called SLIM for Sparse Linear Method) is combined with context similarity in [26] to handle the sparsity problem. A context-aware location recommendation for groups based on the random walk (CLGRW) algorithm is proposed in [27]. Time and weather conditions are also introduced. This approach exploits content similarity and location popularity to reduce sparse data and improve the performance of recommendation. A context-aware collaborative filtering and a knowledge-based approach are combined in [28] to construct a context-aware recommender system for an m-tourism application. This recommendation model provides to a tourist, using a personal mobile device, recommendations on attractions, restaurants, hotels and transportation with regard to a current situation and its profile. [29] proposes a context-aware recommendation approach that uses a combination of novel graph-based similarity with nearest neighbor methods to face the negative effect of sparsity. [4] proposes a novel recommender system, called RSTRC (Recommender System based on Temporal Reliable and Confidence measures), incorporating time factor into reliable and confidence measures. A probabilistic approach is used to evaluate the effectiveness of users' rating profiles and an enrichment mechanism is applied to deal with the sparsity problem by making a denser user-item rating matrix. A temporal similarity measure is used to compute the similarity values of users. [30] proposes two models: CUBCF (Context-similarity User-based Collaborative Filtering recommender model) and CIBCF (Context-similarity Item-based Collaborative Filtering recommender model), using a context-similarity collaborative filtering approach that is exploited to alleviate the sparsity problem of CF models and increase their efficiency. CUBCF is based on user-based collaborative filtering. Split techniques are used to compute the rating-based similarity and chi square kernel is used to compute context-based

similarity. Then, the similarity among users is computed using rating-based and the context-based similarity. In the final step, CUBCF recommends to the active user items with the highest similarity value. CIBCF is based on item-based collaborative filtering. Split techniques are used to compute rating-based similarity and chi square kernel is used to compute context-based similarity. Then, the similarity among items is computed using rating-based and context-based similarity. In the final step, CIBCF recommends items with the highest similarity value. [31] proposes a CARS scheme: CACF (Context-aware Collaborative Filtering) that is based on pre-filtering and post-filtering paradigms with context-weighting approach using Real-coded Genetic Algorithm (RCGA). Further, an Effective Missing Value Prediction (EMPV) algorithm [32] is adopted to handle the sparsity problem. In [33], a non-dominated user neighborhood concept is introduced into DCM (Differential Context Modeling) approaches [34] to develop a CARS framework: ND-DCM (Non-dominated Differential Context Modeling). The non-dominated user neighborhood calculates user-user similarities from multiple dimensions; i.e., the selected similar users have higher similarities with the target user than others from multiple perspectives (e.g. ratings, demographic information, social relationships, …etc.). The DCM approach proposes two context aware recommendation models based on UBCF (User-based Collaborative Filtering): Differential Context Relaxation (DCR) and Differential Context Weighting (DCW). To alleviate the sparsity issue, DCR exploits the optimal-context matching on the rating profiles, while DCW utilizes a weighted context similarity. ND-DCM proposes four approaches: ND-DCR (Non-dominated Differential Context Relaxation), NDs-DCR, ND-DCW (Non-dominated Differential Context Weighting) and NDs-DCW. ND-DCR and ND-DCW measure, respectively, user-user similarities and weighted user similarities based on two matrices: user-item rating matrix (UI) and user-condition rating matrix (UC). NDs-DCR and NDs-DCW measure user similarities based on single rating matrix (UC) with conditions associated with all the selected context dimensions.

Almost all the above mentioned methods are specialized in a specific domain (movies, music, tourism, …etc.), using specific contextual attributes, except CUBCF and the model proposed in [29] which could be applied in any domain. Thus, all the state-of-the-art methods, including these two last models, use splitting techniques to define the target user's neighbors. In this work, similarity is computed in a more simple way; i.e., the similarity is calculated between the target user and all the other users. Then, the users are ordered according to the similarity values and the n first users are defined as the most similar users. Since the authors adopted a user-based approach, R_CARS is compared to the models: CUBCF, ND-DCR, NDs-DCR, ND-DCW and NDs-DCW. The main difference between R_CARS and the other models is that R_CARS adopts a rule-based approach to predict rating, while these models use only context similarity. The objective of this comparison is to illustrate the impact of using rule-based recommendation with a similarity-based collaborative approach on recommendation efficiency.

3. RULE-BASED CONTEXT-AWARE RECOMMENDER MODEL (R_CARS)

In this section, the proposed recommendation model: rule-based context-aware recommender system (R_CARS), is detailed. R_CARS is built based on context-aware collaborative filtering recommendation and rule-based recommendation approach. First, the dataset is preprocessed. Second, R_CARS calculates similarities between the target user and the other users. Finally, the rule-based algorithm is applied on the similar users' dataset in order to generate recommendations (Figure 2).

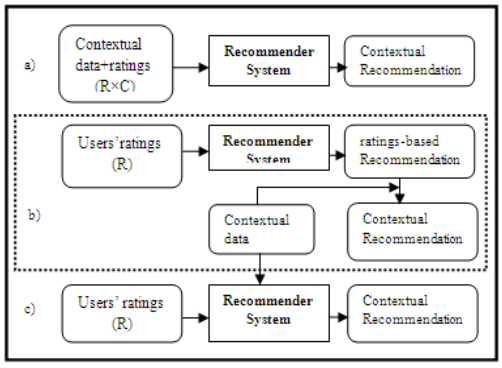


Figure 1. Context-aware recommender models: (a) Pre-filtering, (b) Post-filtering, (c) Contextual modeling.

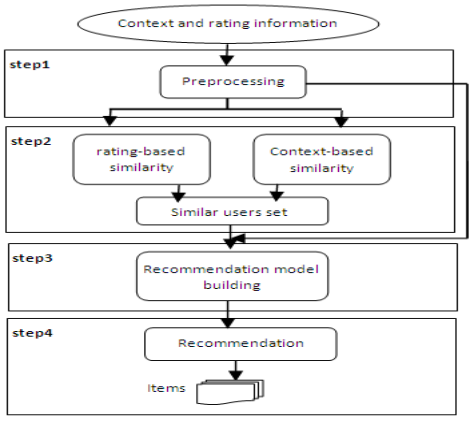


Figure 2. The R_CARS model.

3.1 System Model

The system model used in this work is:
- Let $U=\{u_0,u_1,\ldots u_n\}$, a set of users.
- Let $I=\{i_1,i_2,\ldots i_w\}$, a set of items.
- $R_a=\{r_{a1},r_{a2},\ldots r_{am}\}$ a set of m ratings given by user u_a, with a set of k contextual attributes $C_a=\{c_0,c_1,\ldots c_x\}$. $R_a \subset R$, R is the set of all ratings.
- $P=\{p_{a1}, p_{a2},\ldots p_{aq}\}$ is a set of m predicted ratings to the target user a.
- $L=\{l_1, l_2,\ldots l_z\}$ is a set of induction rules.

3.2 Datasets

In this paper, four real-world datasets are used: DePaulMovie, TheCarMusic, LDOS_CoMoDa and Restaurant.

- DePaulMovie [35]: This dataset contains 5043 ratings provided by 97 users on 319 films, based on three contextual attributes: time, companion and location (the rating scale 1-5).

- InCarMusic [36] contains 4012 ratings provided by 43 users on 139 songs, based on eight contextual attributes: driving style, mood, landscape, …etc. (the rating scale 0-5).

- Restaurant [37] is a public dataset that contains 50 subjects who gave ratings (the rating scale 1-5) to 40 popular restaurants in Tijuana. There are 6 contexts in total which are the combinations of the original two contextual attributes: time and location. In the data, the contexts are represented by S1-S6: S1 (Weekday+school), S2 (Weekend+home), S3 (Weekday+work), S4 (Weekend+school), S5 (Weekend+home), S6 (Weekend+work).

-LDOS_CoMoDa [38] is a public dataset that contains 121 users who rated 1232 movies considering 12 context dimensions (mood, weather, time, location, …etc.). The rating scale 1-5.

3.3 Data Preprocessing

Preprocessing data is the first step in machine learning. The data could contain errors or anomalies that may affect the analysis process, such as: null values, irrelevant features or attributes, inappropriate feature type …etc. Therefore, the data should be preprocessed before applying machine-learning algorithms: classification, regression, clustering, …etc.

Unnecessary attributes are removed, which should not be involved in the recommendation process, such as user Id and item Id. Some other nominal feature types are converted into a string (applying the unsupervised filter: *NominalToString*) before similarity computing. After similarity computing, we reconvert them into nominal (applying the unsupervised filter: *StringToNominal*). Numeric attributes are also converted into nominal, in order to apply the predictive algorithms accurately (using Weka API). For example, time, location, landscape, driving style, rating…etc. In order to improve the prediction results, unsupervised discretization filter (*Discretize*) is used to descritize "rating" into three intervals (categories or classes). Each interval or class presents a rating category. Three categories are defined: the first category C1: [0-2,33] represents non-interesting items; i.e., items having low ratings, while the second category C2:[2,33-3,66] represents items rated with medium values and the third category C3:[3,66-5] represents interesting items (items rated with high values).

3.4 Similarity Computing

Almost all collaborative recommender systems use split techniques in order to define the target user's similar users. These techniques split users into clusters, calculate the distance between the target user and each cluster and then assign the target user to the appropriate cluster. The proposed technique is different and simpler; similarity is computed between the target user and all the other users. Then, the users are ordered according to the similarity values and the n first users are defined as the most similar users. To give flexibility to the experiments, n is selected as 10:5:25 empirically. To calculate the similarity between two users, the authors propose the definition of similarity based on their ratings and similarity based on contextual attributes when these ratings are given. Each user's profile is defined by a matrix with rows as items and columns as ratings and contextual attributes. Each row is composed of ratings and contextual values given on the correspondent item. The users' u_a and u_b profiles are defined

as follows:

$$u_a = \begin{pmatrix} r_{a,1} & c1_{a,1} & c2_{a,1} & ck_{a,1} \\ r_{a,2} & c1_{a,2} & c2_{a,2} & ck_{a,2} \\ r_{a,m} & c1_{a,m} & c2_{a,m} & ck_{a,m} \end{pmatrix} \quad (1)$$

$$u_b = \begin{pmatrix} r_{b,1} & c1_{b,1} & c2_{b,1} & ck_{b,1} \\ r_{b,2} & c1_{b,2} & c2_{b,2} & ck_{b,2} \\ r_{b,m} & c1_{b,m} & c2_{b,m} & ck_{b,m} \end{pmatrix} \quad (2)$$

where $r_{a,m}$ is the rating given by a user u_a on item i_m. The similarity between u_a and u_b is calculated taking into account the items that are co-rated by u_a and u_b. For example, if u_a and u_b rated the items i_1 and i_2, similarity is calculated based on the ratings and contextual values given by the two users on i_1 and i_2. Here, all context attributes are converted from nominal into string for implementation raisons. '*Rating'* is still unchanged (numeric). Formula (3) is proposed to calculate similarity.

$$sim(u_a, u_b) = \sum_{j=1}^{k} (r_{aj} - r_{bj})^2 \times (nb_{mat}/nb) \quad (3)$$

where *sim(u_a, u_b)* calculates the similarity between u_a and u_b. To define ratings-based similarity between u_a and u_b, the sum of the squared differences of their ratings is calculated. r_{aj} is the rating value given by user u_a on item j. r_{bj} is the rating value given by user u_b on item jϵ I. s is the number of items co-rated by users u_a and u_b. The second term calculates similarity based on contextual attributes. The context-based similarity between u_a and u_b is the ratio between the number of match contextual attributes values and the number of available contextual attribute values. nb_{mat} is the number of common contextual attribute values related to items that have been co-rated by users u_a and u_b, whereas nb is the number of all contextual attribute values that user a (or user b) should give. Figure 3 shows how similarity between an active user u_0 and the users: u_1, u_2 and u_3 is computed.

3.5 Rating Prediction

The proposed recommender system is based on one of the data mining techniques: classification. Data mining, also called Knowledge Discovery in Databases (KDD) [39], is the process of discovering interesting information, previously unknown, in a large amount of data involving techniques from machine learning, statistics and database systems [40]. Classification, also called supervised learning, is a data-mining task that assigns items in a dataset to target categories or classes [41, 43]. The first step of classification is the model construction using the training set, which is divided into predefined classes. The model could be built as classification rules, decision trees or mathematical formulae. Model exploitation is the second step, in which new objects from the test dataset, which is independent of the training dataset, will be classified and then the results are compared to the previous step results in order to evaluate the model accuracy [43]. A rule-based classification refers to any classification scheme that makes use of "IF-THEN" rules to predict the class for new items. ZeroR, OneR, PART, JRip, DTNB, ConjunctiveRules, M5Rule, Prism [39]… are examples of rule-based classifiers. Decision trees represents a graph-based classification method that uses tree-like structure to make decisions. A tree is composed of: root node, internal nodes that represent conditions (applied on attributes) and leaf nodes, where leaf nodes' contents represent outcomes or classes. The path between the root node and one of the leaf nodes presents a rule. The rules in decision trees have the form: If (condition1) and (condition 2) and (condition3)…Then outcome [44]. J48, RandomForst, REPTree, RandomTree, NBTree, ADTree, DecisionStump, BFTree… are examples of decision trees.

After identifying the similar users set, data preprocessing is necessary (converting string and numeric attributes into nominal). Then, R_CARS utilizes a classification algorithm to predict ratings (e.g. J48). Next, we explain how predictions are computed using J48.

3.5.1 J48 Classifier

J48 is an open-source Java implementation of the statistical classifier: C4.5 [45]. C4.5 generates a trimmed decision tree from a set of training data and in the same way, the ID3 algorithm does, by using the concept of information entropy. Entropy is a measure of the amount of uncertainty in the dataset *D*. Mathematical representation of entropy is:

$$H(D) = \sum_{cl_i \in CL} -p(cl_i) log_2\, p(cl_i) \quad (4)$$

- D is the current dataset for which entropy is being calculated.
- CL is a set of classes in D.
- p(cl_i) is the proportion of the number of elements in class cl_i to the number of elements in set D.

The entropy allows the computing of the information gain of each attribute. Information gain (a) tells us how much uncertainty in D was reduced after splitting set D on attribute a. Mathematical representation of information gain is:

$$IG(a, D) = H(D) - \sum_{i=1}^{v} p(d_i) H(d_i) \quad (5)$$

- d_i is the subset created from splitting set D by attribute a ($D=\{d_1,\ldots,d_v\}$).
- p(d_i) is the proportion of the number of elements in subset d_i to the number of elements in set D.
- H(d_i) is the entropy of subset d_i.

The J48 algorithm's steps can be defined as follow:

- First, select the attribute with the greatest information gain for the root node and create a new child node for each possible value.
- Then, split the training set into child nodes (subsets).
- Repeat recursively for each child node until getting pure subsets.

3.5.2 J48 for Prediction Computing

In this sub-section, we explain how predictions are calculated using an example from Restaurant Dataset (Figure 4). This example presents the data provided by three similar users. The *rating* attribute is defined as the class attribute. After calculating the information gain of each attribute, the decision tree is built. Then, a set of rules are generated L. To make recommendation for an active user u_0, it is necessary to go down the decision tree and apply the appropriate rule. We suppose that the active user u_0 selects '*applebees'* restaurant under Context S2 (Weekend+home); so, the classifier will predict the rating category: C1, applying the rule l_1.

3.5.3 Alleviating Sparsity

Each data-mining technique has its internal strategy to handle sparsity: Distribution-based Imputation (DBI), Unique Value Imputation (UVI), Predictive Value Imputation (PVI), …etc. [46]. For example, decision trees (C4.5 algorithm) use Distributed-based Imputation (DBI) method to predict missing values [10]. DBI splits the instance to be classified into sub-instances each with different values for missing feature and weight corresponding to the estimated probability of the particular missing value. Each sub-instance is routed down the appropriate tree branch according to its assigned value. Upon reaching a leaf node, the class-membership probability of the sub-instance is measured as the frequency of the class in the training instances associated with this leaf. The overall estimated probability of class membership is calculated as the weighted average of the class membership probabilities over all sub-instances. If there is more than one missing value, the process recurses with the weights combining multiplicatively [46]. For example, suppose attribute a has two possible values: a_1 and a_2; if a node contains 6 branches with $a(?)=a1$ and 4 with $a(?)=a2$, then the probability that $a(?)= a1$ is 0.6 and the

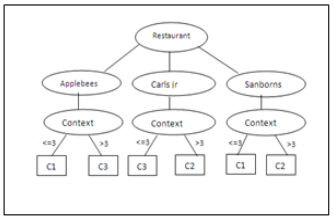
users	items	context	rating
u_1	applebees	1.0	3.0
	burger king	4	2.0
	carls jr	**1**	**5.0**
u_2	**carls jr**	**1.0**	**1.0**
	costco	1	3.0
	burger king	6	1.0
u_3	**burger king**	2.0	5.0
	cafe de la flor	1.0	5.0
	carls jr	**2**	**5.0**

co-rated items

$\Rightarrow$ sim (u_0, u_1)= 1,454
sim (u_0, u_2)= 1,333
sim (u_0, u_3)= 0,576

Figure 3. Similarity computing.

restaurant	context	rating
applebees	2.0	C1
applebees	3.0	C1
applebees	4.0	C3
applebees	6.0	C3
carls jr	1.0	C3
carls jr	2.0	C3
carls jr	3.0	C3
carls jr	5.0	C2
carls jr	6.0	C2
sanborns	1.0	C1
sanborns	2.0	C1
sanborns	1.0	C1
sanborns	5.0	C2
sanborns	6.0	C2



Rules :
l1 : (Restaurant= Applebees) and (context <=3) => rating=C1

l2 : (Restaurant= Applebees) and (context >3) => rating=C3
…

Figure 4. An example of a decision tree (J48).

probability that *a(?)=a2* is 0.4. A fractional 0.6 of (*?*) is now distributed down the branch for a=a1 and a fractional 0.3 of (*?*) down the other tree branch.

3.6 R_CARS Algorithm

The following algorithm summarizes the recommendation model steps:

R_CARS Algorithm

Input: R, I, a given target user u_a.
Output: top k items with highest prediction score to u_a.
Begin
Step 1: Data preprocessing.
Step 2: - Calculate the similarity between u_a and each user of the initial dataset by Eq. (3).
- Construct the top n (n=10:5:25) similar users' dataset.
Step 3: - Data preprocessing.
- Build the recommendation model using the similar users' dataset.
Step 4: - **G**enerate predictions.
- Return the top k items.
End.

4. EXPERIMENT AND FRAMEWORK

To evaluate its effectiveness, R_CAR is implemented and experimented using Weka JAVA API, applying 5-fold cross-validation, on the context-aware datasets: DePaulMovie, InCarMusic, Restaurant and LDOS_CoMoDa, where InCarMusic contains more sparse data than DePaulMovie and LDOS_CoMoDa contains more sparse data than Restaurant. Two experiments are conducted:

- Experiment 1: R_CAR is experimented on DePaulMovie and InCarMusic datasets, introducing four classifiers (JRip, PART, J48 and RandomForest), in order to select the best one compared with CUBCF.

- Experiment 2: R_CAR is experimented on Restaurant and LDOS_CoMoDa datasets, introducing the four classifiers, in order to select the best one compared with ND-DCR, NDs-DCR, ND-DCWand NDs-DCW models.

4.1 Evaluation Metrics

4.1.1 Deviation-based Evaluation Metrics

These metrics, also known as error-based metrics, measure the deviation between the predicted rating value and the real rating value to evaluate the performance of the recommendation algorithm. The deviation-based metrics are: MSE (Mean Squared Error), RMSE (Root Mean Squared Error), MAE (Mean Absolute Error), MAPE (Mean Absolute Percentage Error) and SMAPE (Symmetric Mean Absolute Percentage Error) [47]. Here, MAE and RMSE are used to evaluate the performnce of R_CARS.

$$\text{MAE}=\sum_1^S|raj - paj|/S \quad (6)$$

$$\text{MSE}=\sum_1^S(\text{raj}-\text{paj})^2/S \quad (7)$$

$$\text{RMSE}=\sqrt{\left(\sum_1^S(\text{raj}-\text{paj})^2/S\right)} \quad (8)$$

$$\text{MAPE}=1/S\sum_1^S|\text{raj}-\text{paj}/\text{raj}| \quad (9)$$

$$\text{SMAPE}=1/S\sum_1^S 2|\text{raj}-\text{paj}|/(|\text{raj}|+|\text{paj}|) \quad (10)$$

r_{aj} is the user's a explicit rating to item j. p_{aj} is the user a's predicted rating to item j. S is the number of predicted ratings. The low values of these metrics indicate that the recommendation model is good [48]-[49].

4.1.2 List-based Evaluation Metrics

Lists-based metrics, also known as TOP k recommendation evaluation metrics, verify whether the recommender system is able to recommend interesting items by calculating the rate of interesting items

among those recommended; that is to evaluate the efficiency of the recommendation algorithm using: Precision, Recall and F-measure [48]-[49].

$$\text{Precision} = \text{TP}/\text{TP} + \text{FP} \tag{11}$$

$$\text{Recall} = \text{TP}/\text{TP} + \text{FP} \tag{12}$$

$$\text{F}_{\text{measure}} = 2 * \text{Precision} * \text{Recall}/\text{Precision} + \text{Recall} \tag{13}$$

- TP (True Positives): interesting items recommended by the recommendation model.
- FP (False Positives): uninteresting items recommended by the recommendation model.
- TN (True Negatives): uninteresting items not recommended by the recommendation model.
- FN (False Negatives): available interesting items not recommended by the recommendation model.

4.1.3 Classification Evaluation Metric

In machine learning, the accuracy metric is used to measure the performance of classification algorithms, By comparing the results of the classifier building using the training dataset with the testing classifier results, Accuracy calculates the percentage of items correctly classified [9]. Accuracy is used to evaluate the performance of the four classifiers introduced in this work.

4.2 Experiment 1

The recommendations are presented to an active user u0. The performance of R_CARS is measured using the metrics: MAE, RMSE, Precision, Recall, F-measure and accuracy. These metrics are computed according to the number of similar users (n=10:5:25). The average for each classifier on DePaulMovie and InCarMusic datasets is presented in Table 1 and Table 2, respectively (only the average is presented in this work). In [30], the proposed model: CUBCF, is experimented on the two datasets: DePaulMovie and InCarMusic, applying 5-fold cross-validation and compared with UBCF, IBCF and CIBCF (The results of this experiment are shown in Tables 1 and 2 [30]**)**. A minimal similarity threshold is set to select neighbours.

4.3 Experiment 2

In this experiment, we used the same parameters' values as in the previous experiment (n=10:5:25). Table 3 and Table 4 show the evaluation metrics' values (average) for each classifier on Reastaurant and LDOS_CoMoDa, respectively. In [33], the proposed models: ND-DCR, NDs-DCR, ND-DCWand NDs-DCW, are experimented on the two datasets: Restaurant and LDOS_CoMoDa, applying 5-fold cross-validation (The results of this experiment are shown in Tables 3 and 4 [33]) and compared to some baseline recommendation algorithms: DCW, DCR, CAMF (Context-aware Matrix Factorization), DCW-GA (DCW Genetic Algorithm), …etc. A minimal similarity threshold is set to select neighbours. Only MAE and RMSE are used to compare R_CARS with these models.

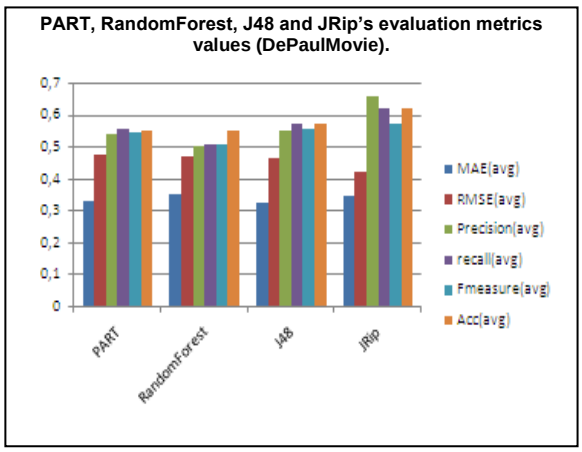


Figure 5. PART, RandomForest, J48 and JRip evaluation metrics' values (DePaulMovie).

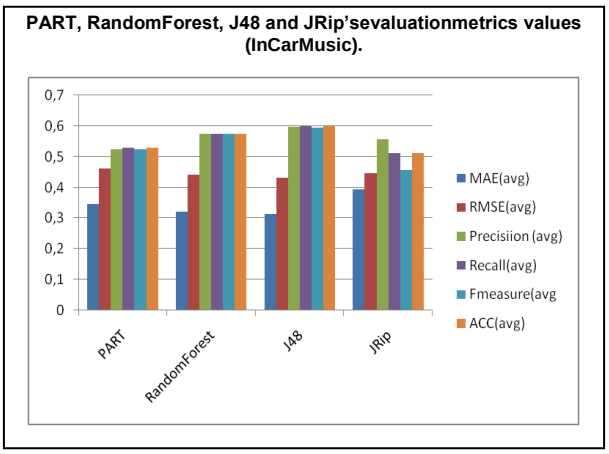


Figure 6. PART, RandomForest, J48 and JRip evaluation metrics' values (InCarMusic).

Table 1. Evaluation metrics' values of the four classifiers (DePaulMovie).

Models	MAE	RMSE	Precision	Recall	F-measure	ACC(%)
R_CAR(PART)	0,327775	0,4713	0,53625	0,55225	0,5425	0,55185
R_CAR(RandomF)	0,348275	0,467375	0,501	0,50775	0,504	0,5075
R_CAR(J48)	**0,3251**	**0,46265**	**0,55**	**0,57025**	**0,55725**	**0,57035**
R_CAR(Jrip)	0,3429	0,4203	0,6595	0,6195	0,571	0,61945
CUBCF	1,39157	1,94572	<0,50	<0,25	<0,25	/

Table 2. Evaluation metrics' values of the four classifiers (InCarMusic).

Models	MAE	RMSE	Precision	Recall	F-measure	ACC(%)
R_CAR(PART)	0,347075	0,460625	0,52375	0,5295	0,52375	0,5291
R_CAR(RandomF)	0,3203	0,44065	0,5735	0,5735	0,5735	0,5747
R_CAR(J48)	**0,31255**	**0,432175**	**0,597**	**0,60075**	**0,59575**	**0,6005**
R_CAR(Jrip)	0,392725	0,44685	0,558	0,513	0,45675	0,51295
CUBCF	1,058	1,28781	< 0,2	< 0,05	< 0,1	/

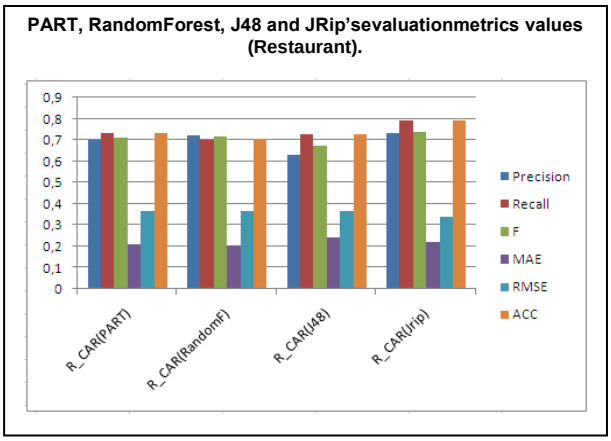


Figure 7. PART, RandomForest, J48 and JRip evaluation metrics' values (Restaurant).

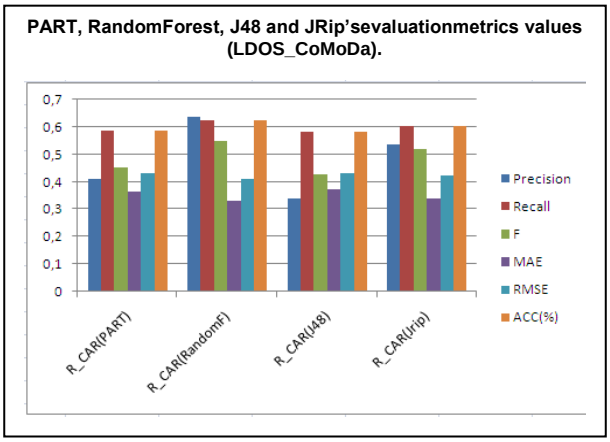


Figure 8. PART, RandomForest, J48 and JRip evaluation metrics' values (LDOS_CoMoDa).

Table 3. Evaluation metrics' values of the four classifiers (Restaurant).

Models	MAE	RMSE	Precision	Recall	F-measure	ACC(%)
R_CAR(PART)	0,210325	0,365425	0,698	0,735	0,71225	0,73475
R_CAR(RandomF)	0,206425	0,36645	0,72475	0,7025	0,715	0,706925
R_CAR(J48)	0,2451	0,364925	0,62975	0,7295	0,6715	0,729675
R_CAR(Jrip)	**0,218625**	**0,340675**	**0,73225**	**0,792**	**0,7365**	**0,792**
ND-DCR	0,784	1,090	/	/	/	/
NDs-DCR	0,787	1,058	/	/	/	/
ND-DCW	0,739	1,002	/	/	/	/
NDs-DCW	0,735	0,997	/	/	/	/

Table 4. Evaluation metrics' values of the four classifiers (LDOS_CoMoDa).

Models	MAE	RMSE	Precision	Recall	F-measure	ACC(%)
R_CAR(PART)	0,36485	0,431725	0,40975	0,58725	0,45375	0,5871
R_CAR (RandomF)	**0,332425**	**0,410875**	**0,63625**	**0,62325**	**0,5495**	**0,6232**
R_CAR(J48)	0,3743	0,430125	0,33875	0,5815	0,428	0,58135
R_CAR(Jrip)	0,34145	0,421425	0,53775	0,60475	0,521	0,60465
ND-DCR	0 ,723	0,939	/	/	/	/
NDs-DCR	0,726	0,938	/	/	/	/
ND-DCW	0,731	0,927	/	/	/	/
NDs-DCW	0, 714	0,927	/	/	/	/

4.4 Result Discussion

In this sub-section, the results obtained in experiment 1 and experiment 2 are discussed. When using DePaulMovie (Table 1), we can note that J48 is the best performing algorithm, having lower error indicators' values and higher values for Precision, Recall, F-measure and accuracy. Most of the results between J48 and JRip are not significant (taking into account all the evaluation metrics). We can also note that R_CARS, introducing the four classifiers, is more efficient than CUBCF (MAE: the difference is over 1,0, RMSE: the difference is over 1,5, Precision: the difference is approximately 0,1, Recall and F: the difference is over 0,3).

Using sparser dataset: InCarMusic (Table 2), we note that J48 outperforms the other algorithms. J48 has lower error indicators' values and higher values of Precision, Recall, F-measure and accuracy, which means that J48 is the best even on datasets with a significant number of sparse data. The results between J48 and RandomForest are not significant. R_CARS, using the four classifiers, can beat CUBCF model (MAE: the difference is over 0,7, RMSE: the difference is over 0,8, Precision: the difference is over 0,3, Recall: the difference is approximately 0,5, F: the difference is over 0,4). CUBCF performance decreases significantly on InCarMusic dataset. This is not surprising since CUBCF depends only on context-based similarity to alleviate sparsity.

When using Restaurant dataset (Table 3), we can see that JRip is the best performing algorithm. The results between JRip and RandomForest are not significant. R_CARS, introducing the four classifiers, can beat the models: ND-DCR, NDs-DCR and ND-DCW (MAE: the difference is over 0,5, RMSE: the difference is approximately 0,7).

Using sparser dataset: LDOS-CoMoDa (Table 4), the experimental results illustrate that RandomForest outperforms the other algorithms, which means that RandomForest is the most efficient on datasets with high level of sparsity. The results between JRip and RandomForest are not significant. Although R_CARS's (introducing the four classifiers) performance decreases, it is still more efficient than those of ND-DCR, NDs-DCR, ND-DCW and. ND-DCW (MAE: the difference is over 0,4, RMSE: the difference is over 0,5). Recall that these models utilize the non-dominated user neighbourhood to alleviate sparsity, while R_CARS combines context-based similarity with data-mining techniques to face this problem.

As a summary, the results demonstrate that RandomForest works better than the other classifiers on datasets with high number of sparse data, while JRip works better on less sparse datasets. In comparison with the state-of-the-art models, R_CARS, taking into consideration the four classifiers, is still the most efficient on the four datasets.

5. CONCLUSIONS

CARSs exploit the knowledge about context under which the ratings are given to generate recommendation. However, in this case, the sparsity issue could be more serious, since there may be limited ratings within the same contextual situation, especially when there are many context dimensions. This paper proposes a novel context-aware recommendation model that we called R_CARS to overcome sparsity issues by combining the proposed user-based similarity technique with a rule-based approach. R_CARS is experimented on four real-world datasets introducing four rule-based algorithms: PART, JRip, RandomForest and J48 and compared to the state-of-the-art models.

The experimental results demonstrate that RandomForest works better than the other classifiers on datasets with a high number of sparse data, while JRip works better on less sparse datasets. The results prove also that R_CARS, introducing the four classifiers, can alleviate the sparsity problem and provide more accurate recommendations compared to the other models. In future work, we will compare the proposed similarity technique with state-of-the-art similarity techniques by using large datasets. We will also work on implementing more effective solutions to overcome the sparsity problem by taking advantage of recent approaches, such as deep-learning techniques [50]-[51].

REFERENCES

[1] S. Benhamdi, A. Babouri and R. Chiky, "Personnalized Recommender System for E-learning Environment," Education and Information Technologies, vol. 22, no. 4, pp. 1455–1477, 2017.

[2] J. Bobadilla, F. Ortega, A. Hernando and A. Gutiérrez, "Recommender Systems Survey," Knowledge-

based Systems, vol. 46, pp. 109–132, 2013.

[3] A. S. Ghabayen and B. H. Ahmed, "Enhancing Collaborative Filtering Recommendation Using Review Text Clustering," Jordanian Journal of Computers and Information Technology (JJCIT) ,vol. 7, no. 2, pp. 152 - 165, DOI: 10.5455/jjcit.71-1609969782, June 2021.

[4] S. Ahmadian, N. Joorabloo, M. Jalili and M. Ahmadian, "Alleviating Data Sparsity Problem in Time-aware Recommender Systems Using a Reliable Rating Profile Enrichment Approach," Expert Systems with Applications, vol. 187, p. 115849, 2022.

[5] G. S. Eshwari and S. P. S. Ibrahim, "Rule-based Effective Collaborative Recommendation Using Unfavourable Preference," IEEE Access, vol. 8, pp. 128116 -128123, 2020.

[6] S. M. Abbes, K. A. Alam and S. Shamshirband, "A SoftrRough Set Based Approach for Handling Contextual Sparsity in Context-aware Video Recommender Systems," Mathematics, vol. 7, no. 8, p. 740, 2019.

[7] Z. Huang, X. Lu and H. Duan, "Context-aware Recommendation Using Rough Set Model and Collaborative Filtering," Artificial Intelligence Review, vol. 35, pp. 85- 99, 2011.

[8] M. Jamali and M. Ester, "Mining Social Networks for Recommendation," Tutorial of ICDM, vol. 11, [Online], Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.367.4838&rep=rep1&type=pdf, 2011.

[9] T. Sanli, C. Sicakyuz and O. H. Yuregir, "Comparison of the Accuracy of Classification Algorithms on Three Datasets in Data Mining: Example of 20 Classes," International Journal of Engineering, Science and Technology, vol. 12, no. 3, pp. 81-89, 2020.

[10] S. F. Huang and C. H. Cheng, "A Safe-region Imputation Method for Handling Medical Data with Missing Values," Symmetry, vol. 12, no. 11, p. 1792, 2020.

[11] E. Kolce and N. Frasheri, "A Literature Review of Data Mining Techniques Used in Healthcare Databases," Proc. of the ICT Innovations, Web Proceedings, pp. 577-582, 2012.

[12] V. Patil and V. B. Nikam, "Study of Data Mining Algorithm in Cloud Computing Using MapReduce Framework," Journal of Engineering, Computers & Applied Sciences (JEC&AS), vol. 2, no. 7, pp. 65-70, 2013.

[13] E. Osmanbegović and M. Suljić, "Data Mining Approach for Predicting Student Performance," Economic Review – Journal of Economics and Business, vol. X, no. 1 , pp. 3-12, May 2012.

[14] I. F. Tobias, P. G. Campos, I. Contador and F. Dies, "A Contextual Modeling Approach for Model-based Recommender System," Proc. of the Conference of Spanish Association for Artificial Intelligence, vol. 8109, pp. 42-51, DOI: 10.1007/978-3-642-406432-0-5, 2013.

[15] V. Codina, F. Ricci and Luigi Ceccaroni, "Distributional Semantic Pre-filtering in Context-aware Recommender Systems," User Modeling and User Adapted Interaction, vol. 26, pp.1–32, 2015.

[16] X. Ramirez-Garcia and M. Garca-Valdez, "Post-filtering for a Restaurant Context-aware Recommender System," Chapter in Book: Recent Advances on Hybrid Approaches for Designing Intelligent Systems, Part of the Studies in Computational Intelligence Book Series, vol. 547, pp. 695–707, 2014.

[17] Y. Zheng, "Deviation-based and Similarity-based Contextual SLIM Recommendation Algorithms," Proc. of the 8th ACM Conference on Recommender Systems (RecSys '14), pp. 437-440, DOI:10.1109/ACCESS.2020.2973755, 2014.

[18] F. Hdioud, B. Frikh and B. Ouhbi, "Multi-criteria Recommender Systems Based on Multi Attribute Decision Making," Proc. of the International Conference on Information Integration and Web-based Applications & Services (IIWAS'13), pp. 203-226, DOI:10.1145/2539150.2539176, 2013.

[19] Y. Zheng, "Context-aware Collaborative Filtering Using Context Similarity: An Empirical Comparison," Information, vol. 13, no. 1, p. 42, DOI: 10.3390/info13010042, 2022.

[20] Y. Shi, M. Larson and A. Hanjalic, "Mining Contextual Movie Similarity with Matrix Factorization for Context- aware Recommendation," ACM Transactions on Intelligent Systems and Technology (TIST), vol. 4, Article no. 16, pp. 1-19, DOI: 10.1145/2414425.2414441, 2013.

[21] Y. Zheng and A. A. Jose, "Context-aware Recommendations *via* Sequential Predictions," Proc. of the 34th ACM/SIGAPP Symposium on Applied Computing (SAC'19), Limassol, Cyprus. pp. 2525-2528, DOI: 10.1145/3297280.3297639, 2019.

[22] H. Al Tair, M. J. Zmerly, M. El Qutayry and M. Leida, "Architecture for Context-aware Pro-active Recommender System," Int. J. Multimedia and Image Proces. (IJIMP), vol. 2, no. 3/4, pp. 125-133, 2012.

[23] F. Rezaeimehr et al., "TCARS: Time and Community-aware Recommender System," Future Generations Computer Systems, vol. 78, no. 1, pp. 419-429, 2017.

[24] M.Unger, A. Bar, B. Shapira and L. Rokach, "Towards Latent Context-aware Recommendation Systems," Knowledge Based System, vol. 104, pp. 165-178, 2016.

[25] L. Liu, N. Mehandjiev and L. Xu, "Using Contextual Information for Service Recommendation,"Proc. of the 44th Hawaii Int. Conference on System Sciences, pp. 1-9, DOI: 10.1109/HICSS.2011.476, 2011.

[26] Y. Zheng, B. Mobasher and R. Burke, "Similarity-based Context-aware Recommendation," Proc. of the International Conference on Web Information Systems Engineering (WISE 2015), Part of the Lecture Notes in Computer Science Book Series, vol. 9418, DOI: 10.1007/978-3-319-26190-4_29, 2015.

[27] E. Khazei and A. Alimohammadi, "Context-aware Group Oriented Location Recommendation in Location-based Social Networks," ISPRS International Journal of Geo-information, vol.8, no. 406, 2019.

[28] A. V. Smirnov et al., "Group Context-aware Recommendation Systems," Scientific and Technical Information Processing, vol. 41, no. 5, pp 325-334, 2014.

[29] T. M. Phuong, D. T. Lien and N. D. Phuong , "Graph- based Context-aware Collaborative Filtering," Expert Systems with Applications, vol. 126, pp. 9-19, DOI: 10.1016/j.eswa.2019.02.015, 2019.

[30] H. X. Hiep et al., "Context-similarity Collaborative Filtering Recommendation," IEEE Access, vol. 8, pp. 33342-33351, DOI: 10.1109/ACCESS.2020.2973755, 2017.

[31] S. Linda, S. Minz and K. K. Bharadwaj, "Effective Context-aware Recommendations Based on Context Weighting Using Genetic Algorithm and Alleviating Data Sparsity," Applied Artificial Intelligence, vol. 34, no. 10, pp.730-753, DOI: 10.1080/08839514.2020.1775011, 2020.

[32] H. Ma, I. King and M. R. Lyu, "Effective Missing Data Prediction for Collaborative Filtering," Proc. of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.39-46, 2007.

[33] Y. Zheng, "Non-dominated Differential Context Modeling for Context-aware Recommendation," Applied Intelligence, vol. 2022, DOI: 10.1007/s10489-021-03027-5, 2020.

[34] Y. Zheng, R. Burke and B. Mobasher, "Differential Context Relaxation for Context-aware Travel Recommendation," Proc. of the 13th International Conference on Electronic Commerce and Web Technologies (EC-WEB), vol. 123, pp. 88-99, 2012.

[35] Github, "Movie_DePaulMovie," [Online], Available: https://github.com/irecsys/CARSKit/blob/master/context-aware_data_sets/Movie_DePaulMovie.zip.

[36] Github, "Music_InCarMusic," [Online], Available:https://github.com/irecsys/CARSKit/blob/master/context-aware_data_sets/Music_InCarMusic.zip.

[37] X. R. Garcia and G. M. Valdz, "Post-filtering for a Restaurant Context-aware Recommender System," Proc. of Recent Advances on Hybrid Approaches for Designing Intelligent Systems, Part of Studies in Computational Intelligence, vol. 547, pp. 695-707, 2014.

[38] A. Kosir, A. Odic, M. Kunaver, M. Tkalcic and J.F. Tasic, "Database for Contextual Personalization," Elektrotehn Vestnik, vol. 78, no. 5, pp. 270-274, 2011.

[39] B. Sunita, L. M. Aher and R. J. Lobo, "Data Mining in Educational System Using WEKA," Proc. of the International Conference on Education and Training Technologies (ICETT 2011), no. 3, pp. 20-25, 2011.

[40] K. Saravananathan and T. Velmurugan, "Analysing Diabetic Data Using Classification Algorithms in Data Mining," Indian Journal of Science and Technology, vol. 9, no. 43, pp. 1-9, November 2016.

[41] D. Chang, J. Liu, Z. Xu, H. Li, H. Zhu and X. Zhu, "Context-aware Tree-based Deep Model for Recommender Systems," Proc. of DLP-KDD, ACM NY, USA, DOI: 10.48550/arXiv.2109.10602, 2021.

[42] B. Sunita, L.M. Aher and R.J. Lobo, "Comparative Study of Classification Algorithms," International Journal of Information Technology and Knowledge Management, vol.5, no. 2, pp. 239-243, 2012.

[43] N. Bharagava et al., "Decision Tree Analysis on J48 Algorithm for Data Mining," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 3, no. 6, 2013.

[44] E. Ahishakiye, E.O Omulo, D. Taramwa and I. Niyonzima, "Crime Prediction Using Decision Tree (J48) Classification Algorithm," International Journal of Computer and Information Technology, vol.6, no. 3, pp. 188-195, 2017.

[45] J. R. Quinlan, "Induction of Decision Tree," Machine Learning, vol. 1, pp. 81-106, 1986.

[46] M. S. Tsechansky and F. Provost, "Handling Missing Values When Applying Classification Models," Journal of Machine Learning Research, vol. 8, pp. 1625-1657, 2007.

[47] N. Q. Phan, P. H. Dang and H. X. Huynh, "Statistical Implicative Similarity Measures for User-based Collaborative Filtering Recommender System," Int. J. of Advanced Computer Science and Applications (IJACSA), vol. 7, no. 11, pp.140-146, 2016.

[48] A. Gunawardana and G. Shani, "A Survey of Accuracy Evaluation Metrics of Recommendation Tasks," Journal of Machine Learning Research, vol. 10, pp.2935–2962, 2009.

[49] J. L. Herlocker, J. A. Konstan et al., "Evaluating Collaborative Filtering Recommender Systems," ACM Transactions on Information Systems, vol. 22, no. 1, pp.5-53, 2004.

[50] B. Ouhbi et al., "Deep Learning Based Recommender Systems," Proc. of the 5th IEEE International Congress on Information Science and Technology (CiSt), pp. 161-166, DOI: 10.1109/CIST.2018.8596492, Marrakech, Morocco, 2018.

[51] S. Ahmadian, M. Ahmadian and M. Jalili, "A Deep Learning Based Trust-and Tag-aware Recommender System," Neurocomputing, vol. 488, pp. 557-571, DOI: 10.1016/j.neucom.2021.11.064, 2021.

ملخص البحث:

يُعدّ التنائر مشكلةً جديّةً في التّصفية التعاونية التي لها أثر ملحوظ في جودة التّوْصية. ويتمّ إدخال المعلومات السّياقية في أنظمة التّوْصية التّقليدية الى جانب معلومات المستخدِمين ومعلومات المفرَدات للتّغلّب على هذه المشكلة. وقد أثبت العديد من الأعمال البحثية أنّ إدخال المعلومات السّياقية قد يزيد من البيانات المتناثرة. لذا فإنّ تقنيات تعدين البيانات هي من بين الحلول الفعّالة التي جرى استخدامُها في أنظمة التّوصية الواعية للسّياق للتّعامل مع مشكلة تناثُر البيانات.

تقترح هذه الورقة دمْج تقنية توصية للتّصفية التّعاونية تقوم على تشابُه السّياقات والمستخدمين مع تقنيات تعدين البيانات كحلّ لهذه المشكلة من أجل تطوير نظام توصية مبتكر، وهو نظام مبني على القواعد وواعٍ للسّياق. وقد تمّت تجربة النّظام المقترح بإدخال أربع خوارزميات مبنية على القواعد على أربع مجموعات بياناتٍ مختلفة، ومن ثمّ مقارنة أداء النّظام المقترح بأداء النّماذج الواردة في أدبيات الموضوع. وقد بينت نتائج التّجربة أنّ وزْن التّشابُه مع السّياق ودمجه مع تقنية مبنية على القواعد من الممكن أن يقودا الى التّغلّب على مشكلة تناثُر البيانات ويعملا على تحسين دقّة التّوصية بالمقارنة مع النّماذج الواردة في أدبيات الموضوع.