INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2024, VOL. 70, NO. 1, PP. 103–114
Manuscript received November 17, 2023; revised March, 2024. doi: 10.24425/ijet.2024.149520

Selection of clusters based on internal indices in multi-clustering collaborative filtering recommender system

Urszula Kużelewska

***Abstract*—The successful application of a multi-clustering-based neighborhood approach to recommender systems has led to increased recommendation accuracy and the elimination of divergence related to differences in clustering methods traditionally used. The Multi-Clustering Collaborative Filtering algorithm was developed to achieve this, as described in the author's previous papers. However, utilizing multiple clusters poses challenges regarding memory consumption and scalability. Not all partitionings are equally advantageous, making selecting clusters for the recommender system's input crucial without compromising recommendation accuracy.**

This article presents a solution for selecting clustering schemes based on internal indices evaluation. This method can be employed for preparing input data in collaborative filtering recommender systems. The study's results confirm the positive impact of scheme selection on the overall recommendation performance, as it typically improves after the selection process. Furthermore, a smaller number of clustering schemes used as input for the recommender system enhances scalability and reduces memory consumption. The findings are compared with baseline recommenders' outcomes to validate the effectiveness of the proposed approach.

***Keywords*—multi-clustering; collaborative filtering; evaluation of clustering schemes**

I. Introduction

RECOMMENDER systems (RSs) emerged as a response to the accelerated expansion of the Internet and, consequently, a large increase in distributed data. They are electronic applications to help users reach the information or resource they are interested in, quickly and conveniently. Usually, their outcome is collected in the form of a list of recommender items, typically ranked, which is presented to users [1], [2].

Recommender systems are universal enough to be applied to various problems from different domains. The most common are internet stores, with the originator, Amazon, as well as music and movie services, e.g, LastFM or Netflix with their original recommenders: respectively, Audioscrobbler [3] and Cinematch [4]. Other examples are: internet news servers [5], travel attractions or hotels [6], resources for e-learning [7], food [8] and photos [9].

Considering the type of input data and base methods used to generate recommendations, recommender systems are divided into content-based, collaborative filtering (CF) and knowledge-based [10]. Content-based approach (also called content-based filtering) utilizes attribute (characteristic) vectors of items created from text. The text is connected to the items, e.g., it comes from their description, including all additional information, such as a title or an author. Knowledge-based methods are better for one-time users, e.g., in stores selling cameras (people do not often buy cameras). The approach uses both technical attributes of the items as well as user preferences. The attributes are often weighted.

Collaborative filtering techniques are a very attractive solution in the domain of RSs [11], [2]. They search for similarities among users (user-user approach) or items (item-item approach). The system considers users' historical data, including their search records, internet browsing history, and rated items. Collected data are furtherly used for searching similarity among users, with an underlining supposition that users with comparable activity select the same items. As a consequence, recommenders are able to evaluate the level of interest of those users on different items which are new to them [12]. Collaborative filtering methods are particularly recognized for generating accurate recommendations [13].

Although many novel techniques, which are complex and sophisticated, to generate proposition lists were proposed by researchers, there is still a demand to design a comprehensive recommender system that is accurate, scalable, and time efficient [12]. Moreover, they involve coping with high data sparsity regardless of its size. It is expressed by both vertical and horizontal scalability. Vertical scalability corresponds with providing recommendation lists in real-time despite data size, whereas horizontal scalability is related to data sparsity [10].

Clustering algorithms are attractive tools for addressing the vertical scalability problem [14]. They identify groups of similar items (or users) that can contribute to recommender systems for *a priori* identification of neighbourhood objects related to a target one (e.g., a target user is a user to whom recommendations are generated). Clustering algorithms, on the other hand, have their weak points as well. First of all, most

The work was supported by a grant from the Bialystok University of Technology WZ/WI-IIT/3/2023 and funded with resources for research by the Ministry of Education and Science in Poland.

U. Kużelewska is with Faculty of Computer Science, Bialystok University of Technology, Wiejska 45a, 15-351 Bialystok, Poland (e-mail: u.kuzelewska@pb.edu.pl).

of them have input parameters, which different values highly influence the final results. Moreover, the outcomes can differ even though the values remain the same. It is related to the way how they work - their purpose is not to find a globally optimal partition but a local one, starting with different initial points [15]. Different clustering schemes affect the accuracy of recommendations generated by recommender systems due to changes in the neighbourhood range of target objects.

The challenges identified above can be accompanied by new methods based on clustering. They are described as multiple clusterings, multi-clustering, or ensemble clustering [16], [17]. They can vary in detail; however, their common idea is to implement multiple runs of clustering algorithms or to apply multiple applications of a partitioning process on different input data. In [17], the authors used a general expression of $multiple\ clusterings$ and divided the algorithms into the following types: ensemble clustering, clustering with multiple criteria, distributed clustering and three-way clustering.

The Multi-Clustering Collaborative Filtering algorithm, as described in [18], operates with multiple clustering schemes instead of relying on a single one. These schemes are generated through multiple runs of a clustering method, each using different values of an input parameter. Such an operation fulfils the ensemble clustering purpose, which is "to find a combined clustering result based on multiple clusterings of the dataset" [17]. The clustering method utilized in this study is $k-means$, which is a widely used and comprehensive partitioning solution. It was employed with varying values for the number of clusters to generate different clustering schemes for the Multi-Clustering Collaborative Filtering algorithm.

The first experiments, which were described in [18], validated $M-CCF$ against baseline predictors: an item-based and a single-clustering recommender systems. The item-based approach ($IBCF$) identifies neighbourhood using k Nearest Neighbours algorithm and the single-clustering techniques ($SCCF$) utilize only one partitioning scheme for this purpose. The advantage of $M-CCF$ was gained in terms of recommendation quality. Unfortunately, other aspects of performance worsen, specifically the time it took to generate recommendations, especially when compared to the solution using only one clustering scheme as input. It is related to an additional amount of time required to select one of the clustering schemes. Moreover, retaining multiple clustering schemes as input results in high memory consumption, which negatively impacts the algorithm's scalability.

The first idea of selecting the clusters that $M-CCF$ includes as a data model was presented in [19]. This paper extends the experiments to include a larger dataset, comparing the results against the abundance and parameters of the dataset, and allowing generalization of the conclusions.

Indicating the particularly effective clustering schemes reduces the number of clusters to forward on $M-CCF's$ input. As selection tools, internal indices were used, to measure the level of compactness and separability of the clustering schemes. It was decided to measure the systems' accuracy as it is an elementary indicator for comparing their performance. Although there are many other indices, such as diversity, serendipity, and novelty, they are used to evaluate systems' general performance, as a supplement to accuracy.

Our main contributions are as follows:

- Selection of clusters to forward to $M-CCF$ input is beneficial for its performance in terms of recommendation accuracy and coverage, measured respectively by $RMSE$ and $Coverage$.
- Criteria based on internal indices to assess the quality of clustering schemes is an appropriate approach to identify valuable clusters, regardless of data size or density, and thus to provide selected clusters to $M-CCF$ input.

The article is structured as follows: the following section provides an overview of the neighborhood identification problem using clustering algorithms within the context of Recommender Systems. This section also explores various existing solutions along with their respective advantages and disadvantages.

Moving on, Section III presents the proposed multi-clustering algorithm, named $M-CCF$, in comparison to alternative clustering techniques. Subsequently, Section IV focuses on the cluster selection procedure.

The subsequent section presents the results obtained from conducted experiments, comparing the multi-clustering approach with baseline algorithms. The baseline algorithms include a standard item-based collaborative filtering method based on the kNN technique ($IBCF$), and a system employing a single-clustering scheme ($SCCF$).

Finally, the last section concludes the paper, summarizing the findings and implications of the study.

II. Background and Related Work

Although many new, often complex, solutions are proposed, the approaches based on neighbourhood determination are still intensively developed and often outperform other algorithms [20], [21].

A baseline method for neighbourhood identification is k Nearest Neighbours (kNN). It works as follows: it searches for the most similar k users (or items) to the target one and determines them as the target neighbourhood. It must be admitted, that all user-user or item-item similarities must be calculated for this purpose. Nonetheless, the processing time is reduced due to further calculations being performed on a smaller neighborhood size. This reduction in size is a result of employing multiple clustering schemes and selecting only relevant clusters for processing. This method is a reference approach used to neighbourhood calculation in recommender systems based on collaborative filtering [13]. An example application is [22], a genetic-based recommender system, in which the neighbourhood was hybridized with the latent factor models. Experiments with this technique and $M-CCF$ are described in [23].

The algorithm based on k Nearest Neighbours is an attractive solution due to its simplicity and reasonably accurate results. Nevertheless, it has certain disadvantages, as well. The main disadvantages are: low scalability and susceptibility to data sparsity issues [12].

The answer to these challenges can be clustering algorithms due to the way of neighbourhood identification. This process is executed once for all data and in an offline mode. As a result, all cluster members belong to one neighbourhood area.

Although clustering algorithms improve the efficiency of the neighbourhood identification process, they introduce other problems to solve. They are the following: a considerable decrease in prediction accuracy and diversity in recommendation results. The second issue is related to the fact that many popular clustering methods are non-deterministic and therefore their results, which are clustering schemes, can diverge from each other notwithstanding the same values of input parameters. The following section, Section II-A includes an explanation of this phenomenon and its impact on recommendations.

The multi-clustering approach, instead of one partitioning scheme, performs on a set of clusterings, therefore the most relevant can be selected in order to model the neighbourhood. Hence, the negative effect of non-determinism can be eliminated. Section III describes $M-CCF$ algorithm with the advantages of the multi-clustering application. This algorithm works as the item-item approach, therefore the items are clustered.

A. Clustering Approach to Neighbourhood Identification

Generally, several problems need to be considered to achieve high-quality clustering. First of all, a final partitioning is determined by the values of parameters given to the algorithms' input. It means that the discovery of optimal values of these parameters is essential but demanding. Moreover, evaluating clusters to select for the recommendation process is also challenging. Furthermore, some partitioning schemes may be more appropriate for some particular applications, whereas others operate more excellently in other solutions [24]. The additional problem of using clusters in recommender systems is decreasing prediction accuracy. It results from incorrect neighbourhood identification of the data located on the borders of groups. In these circumstances, the objects from other groups located close to the border data may appear to be more similar to the active user. This occurs due to the influence of neighboring clusters and the selection process, potentially affecting the perceived similarity between the user and certain items. It is discussed in detail in [18].

Indeed, a selection procedure for the clustering algorithm is crucial. It plays a vital role in determining the most relevant and effective clustering schemes to be utilized in the recommender system. Simplicity and high scalability make $k-means$ one of the most popular clustering techniques [15], particularly useful in collaborative filtering recommender systems. A variant of $k-means$ algorithm, bisecting $k-means$, was proposed for privacy-preserving applications [25]. The authors in [26] applied $k-means$ for items clustering in a movie recommender with online learning automata-based user profiling. As a result, the final accuracy of recommendations increased. Another method, ClustKNN [27] was used to handle large-scale RS applications. Two-stage clustering was applied in [28] to implement a concept of so called $RatingBubbles$. They appear when users and items are grouped into homogeneous clusters. An interesting algorithm, BICE, was proposed in [29]. It uses a bio-inspired ensemble clustering method exploiting Mussels Wandering Optimization (MWO) and Particle Swarm Optimization for travel recommendations. In [30], a biclustering approach, with clusters' overlap, is used for neighbourhood formation. The authors obtained a strong partial similarity with active users' preferences. One of the recent solutions [31] applies hierarchical clustering to extract clusters from a hierarchy of candidates automatically. It can be applied as a preprocessing step in an arbitrary recommender system. The method does not require critical parameters, such as a number of clusters and directly minimizes data sparsity within the groups. Bi-MARS algorithm [32] was particularly efficient for the precise neighbourhood identification of a target user. The authors employed a bi-clustering approach by creating a lattice of bi-clusters to find the closest neighbourhood of similar users. The algorithm benefited prediction accuracy and was both vertically as well as horizontally scalable.

Nevertheless, the $k-means$ solution, as well as many other clustering methods, do not always guarantee clustering convergence. Moreover, they require input parameters to be given, e.g., a number of clusters, as well. Typically, different initialization points affect the different results of partitioning [33].

III. Presentation of M-CCF Algorithm

The proposed method performs on several types of clustering schemes that are delivered for $M-CCF$ algorithm's input. It is implemented in the following way (for the original version, with one type of a clustering scheme, check in [18], [23]).

Step I. Multiple clustering

The first stage of the $M-CCF$ algorithm is to identify clusters in the input data. In the experiments presented below, the items were clustered - it is explained in the following section. The process is repeated several times and all outcomes (for comparison - in a traditional single-clustering, only one partitioning is generated) are saved in order to transfer them to $M-CCF$. In the experiments described in this paper, $k-means$ was selected as a clustering algorithm, which was executed for $k=5,20,50,100$ to generate the schemes (denoted by C set) for one $M-CCF$ RS system. This is illustrated in Figure 1 presenting both versions of $M-CCF$. On the left side the old version is illustrated - several clustering schemes, but with the same value of k, are transferred to the RS. The right side presents the new variant of $M-CCF$, which was used in the experiments described below - several clusterings are sent to the algorithm as well, however, the value of k was diverse.

Step II. Building M-CCF RS system

It is an essential concern to have accurate neighbourhood modelling for all data objects. It is performed by iterating every input object and identifying the most appropriate group from C set for it. In the case of items clustering, every item needs to have the most appropriate cluster identified. The

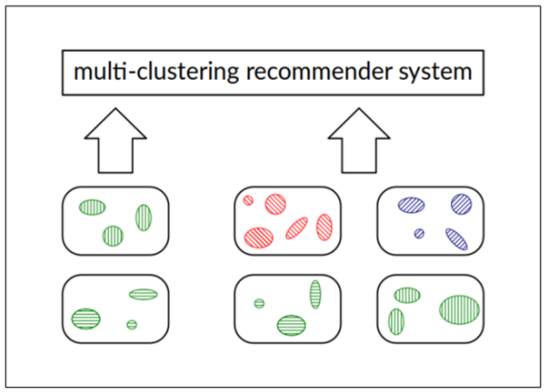

Fig. 1. Comparison of different inputs in $M-CCF$ algorithm

term *the most appropriate* is related to the cluster whose center object is the most similar to the particular input data. Then, when all input objects have their associated clusters, traditional separate CF systems are built on these clusters, which provide input data sets for them. Consequently, $M-CCF$ system is built - an aggregate of recommender algorithms created on particular clusters.

Step III. Recommendation generation

First of all, a relevant RS from $M-CCF$ is selected, when recommendations for a target user are generated. It also utilizes the similarity between the target user's and cluster centers' ratings. Then, the process of recommendation generation is executed as it is implemented in the traditional collaborative filtering approach. However, searching for similar items is restricted to the group connected to the particular recommender in $M-CCF$ algorithm.It means that if a target user needs to evaluate a particular item's rating, the item's similarity to other items is searched only within the cluster to which the particular item belongs.

In this case, when a neighbourhood is represented by a single-clustering scheme, the items or users located on the border of clusters have fewer neighbours in their nearby area than the ones located in the middle of a group. Moreover, if the clusters are located close to one another, it may occur that more similar neighbours are the ones belonging to the other clusters.The multi-clustering avoids such situations, as it recognizes clusters where particular users or items are very close to its center. A more detailed description of this phenomenon is presented in [18], [23].

A major advantage of $M-CCF$ algorithm is the better quality of an active user's neighbourhood modelling, therefore resulting in high precision of recommendations, including sparse cases. However, multiple clustering schemes to keep on the recommender system's input require more memory and additional time for the step of appropriate cluster identification. Moreover, some schemes are not used in the recommendation process as long as they do not contain the optimal location of objects. It is worth examining whether the selection procedure is viable and whether the criteria based on internal indices' value improve the performance of $M-CCF$.

IV. Cluster Selection Techniques

Each clustering algorithm is distinguished by its own strengths and weaknesses. On a particular data set, different methods or the same algorithms with different input parameter values usually lead to various results. To address this issue, a concept of *cluster ensemble* or *clustering aggregation* emerged to integrate several partitonings into a final outcome [34]. One of the approaches to this concept that generates a set of base clustering schemes is to run a single clustering algorithm with different initial sets of parameters several times [35]. Then, a cluster selection procedure can be applied to determine the relevant ones to a particular problem.

Cluster ensembles are widely used in data mining tasks, including recommendation generation. In [35] a recommender system is proposed, which uses $k-means$-based method called $KMCE$, to select a final result from many base clustering schemes. It evaluates the local credibility of each cluster label, building the relation between clusters, and generating the final outcome. The authors used Rand index as one of the evaluation criteria. Recommendation accuracy was raised in [14] by applying a combination of PCA and $k-means$ methods. The authors used $Dunn$ index to evaluate clusterings. In [36], the authors used $k-means$ and, to avoid convergence in clustering results, applied a procedure of initial centroid selection, which discovered underlying data correlation structures. They compared the proposed solution to base one, where cluster centers are randomly initialised. As a result, recommendation accuracy and coverage were improved.

Evaluation of clustering algorithms' performance is not trivial due to the lack of group labels and precisely formulated objectives. In some cases, if the labels can be delivered for evaluation, it is possible to use them in so-called external indices, e.g. Rand index, Fowlkes-Mallows score [15]. If they are unavailable, the only option is to use internal measures exploiting similarity among data objects [15]. In the experiments described below, the following measures were applied: Silhouette (SH) [37], Davies-Bouldin (DB) [38] and Calinski-Harabasz (CH) [39].

The Silhouette Coefficient contains 2 main components: a and b, which are defined by (1):

$$\begin{aligned} SH &= \forall_{y_i \in Y} \overline{SH(y_i)},\ SH(y_i) = \frac{a(y_i) - b(y_i)}{\max(a(y_i), b(y_i))} \\ a(y_i) &= \frac{1}{\|C_i\| - 1} \sum_{j \in C_i, i \neq j} d(y_i, y_j) \\ b(y_i) &= \min_{k \neq i} \frac{1}{\|C_k\|} \sum_{j \in C_k} d(y_i, y_j) \end{aligned} \tag{1}$$

The component $a(y_i)$ calculates an average distance between y_i (data point belonging to the dataset Y) object and all other objects in the same cluster C_i, whereas $b(y_i)$ is an average distance between y_i object as well, but all other objects in the nearest cluster C_k.

The range of SH is [-1,1], with naturally correct clusters was identified by higher values. It better works on separable clusters, otherwise its scores are around 0.

The Davies-Bouldin coefficient is described by (2):

$$DB = \frac{1}{\|C\|} \sum_{C_i, C_j \in C} \max_{i \neq j} \frac{diam_{C_i} + diam_{C_j}}{D(C_i, C_j)}$$
$$diam_{C_i} = \forall_{k \in C_i} \overline{d(c_i, k)},\ diam_{C_j} = \forall_{m \in C_j} \overline{d(c_j, m)}, \quad (2)$$
$$D(C_i, C_j) = d(c_i, c_j)$$

The components $diam_C$ stand for a cluster diameter, which is an average distance between the cluster center c_j and all other cluster's points, whereas D is a distance between clusters, calculated by the distance between their centers (c_i, c_j). The required value is around 0 that is related to well separable clusters.

This index quantifies the average dispersion between clusters and within clusters, providing valuable insights into the quality of the clustering solution. A lower value suggests better separability and more distinct clusters, which is desirable in clustering analysis. When the Calinski-Harabasz index approaches 0, it indicates that the clusters are well-defined and the items within each cluster share common characteristics, making them distinguishable from items in other clusters. That means it is suitable for separable and convex clusters. It is calculated as follows (3):

$$CH = \frac{Sep(C)}{Coh(C)},\ Sep = \frac{\sum_{C_i \in C} \|C_i\| \cdot d(c_i - c)}{\|C\| - 1},$$
$$Coh = \frac{\sum_{C_i \in C} \sum_{j \in C_i} d(y_j, c_i)}{\sum_{C_i \in C} (\|C\| - \|C_i\|)} \quad (3)$$

CH value is related to similarity of objects to their own clusters (cohesion - Coh) compared to other clusters (separation - Sep). The $d(c_i - c)$ component in Sep calculates distance between the cluster center c_i and the global centroid c. The $d(y_j - c_i)$ component in Coh calculates distance among the cluster center c_i and the other cluster points y_j. The highest values of CH mean better result partitions.

The indices described above were applied to evaluate clustering schemes generated by $k-means$ algorithm in order to eliminate the useless ones. The detailed idea and a procedure of application of this index in the process of clusters' selection are described in Section V-A.

V. Experiments

The experiments were performed with the purpose to verify whether the selection of clusters based on internal indices is efficient in $M-CCF$ recommender system. In other words, whether the performance of $M-CCF$ is improved when the schemes negatively evaluated by the indices are removed from the input data.

The experiments were divided into 2 phases: clustering with clusters' evaluation and generating recommendations with a measurement of their accuracy. In the first phase, $k-means$ was taken as it is the most common clustering algorithm and was successfully deployed in the previous version of $M-CCF$ approach [18]. The final accuracy is evaluated against both $SCCF$ and $IBCF$ recommender systems.

Two subsets of the MovieLens dataset [40] were taken for this purpose. Originally, the data contained 25 million ratings, however, in the experiments, randomly selected samples were taken. The subsets are presented in Table I: the small one, consisted of 100 000 ratings (100k) and a big one composed of 1 million ratings (1M). Both datasets were split into training and testing parts in the proportion of about 100 to 1. Note, that the small set is more sparse than the big one - it contains a smaller number of ratings per item: 9.06 in comparison with 60.49.

A. Clustering and Evaluation of Clustering Schemes

The clustering process was executed several times with the following values of k: 5, 20, 50 and 100. The numbers were selected during many experiments as having the most impact on the recommendation accuracy. Furthermore, it used various distance measures: cosine-based, Euclidean, CityBlock and Tanimoto-based. It was decided to cluster the items (movies) - for this reason, an item-item recommender system was selected to use in the following phase. The opposite version - users' clustering - was also examined. However, the problem arose in the groups' content. They were composed of one great cluster that contained about 50% of data, and many very small ones with many users remained nonclustered.

Every run of $k-means$ was repeated 6 times, and each result (a clustering scheme) was evaluated regarding compactness and separability. The following indices were taken for this purpose: Silhouette (S), Davies-Bouldin (DB) and Calinski-Harabasz (CH). The implementation in Python's Scikit Learn library was applied [41]. Figures 2 and 3 present evaluation results on the small and big datasets respectively, for each of the 6 schemes. The best clustering schemes were utilized as input sets for $M-CCF$ algorithm in the second phase of the experiments.

During the evaluation of clustering schemes, the values of internal indices were analysed. Both indices, Silhouette and Calinski-Harabasz, indicate the best scheme with the highest values, with Silhouette not exceeding 1. Whereas, Davies-Bouldin index's desired value is the lowest, around 0. Although all of them can detect well separable and compact clusters, the evaluation process was not a straightforward task. In numerous cases, the evaluation values of particular indices were not considerably diversified to imply an appropriate result, and additionally, the indices were not consistent. In the definitive selection, the following rules were applied: the importance of a level of difference in every particular index's value and voting of the indices in the case of inconsistency.

Figure 2 presents evaluation results on the $100k$ dataset. The left top graph's data was clustered with cosine-based distance. The solid line (identically on all graphs) denotes an assessment of 5 groups and has a clear appointment of all indices with the 2nd and 5th scheme as the best results. The dashed line denotes the evaluation of 20 groups and both Calinski-Harabasz and Silhouette indices indicate their 5th scheme as the best result, whereas the Davies-Bouldin index has the lowest value for the 6th clustering, but the difference between the 5th and 6th result is very slight. Both evaluation lines - CH and S

- for 50 group schemes (dotted lines) are rather flat, which means they were not able to distinguish any result opposed to DB, which indicates its 5th clustering as the best one. A similar situation is observed in the case of 100 group schemes (dash-dotted lines) in which all indices' graphs are rather equal with extremely slight fluctuations on both schemes: the 2nd and 4th. Finally, the following schemes were selected for a recommendation phase: the 2nd, the 5th (5 groups), the 5th, the 6th (20 groups), and the 5th (50 groups).

The right top graph in the same figure (Figure 2) presents analogous results, however, a distance measure during the clustering process was Euclidean. These schemes are more difficult to select due to the lack of very distinct points on the evaluation graphs. However, in order to do further research in this space complete, some schemes were selected in this case, as well: the 3rd, the 5th (5 groups), the 2nd, the 4th (20 groups), the 6th (50 groups), the 1st, the 2nd, the 3rd (100 groups) due to slightly better indices' performance.

Corresponding results, but for a CityBlock distance as a clustering metric, are shown on the left bottom graph in the same figure. The results were evaluated unambiguously for the 5 group clusterings, with the 3rd and 4th schemes specified as the best. In the following case, the results of the 20 group clusterings were rather clear with the 1st and 3rd schemes selected. For the 50 group clusterings, only the 1st scheme was taken, however, the values of indices were not very unambiguous. The final case, the set of 100 group clusterings, was evaluated unanimously and the following schemes were chosen: the 3rd, the 5th.

Figure 3 shows evaluation graphs for clusters obtained on the big dataset - $1M$. In the case of a cosine-based clustering metric, the following schemes were selected. When $k-means$ had $k = 5$, the best clusterings were the 1st and the 6th. Indeed, the Silhouette index was the lowest for the 6th scheme, but considering its nearly flat line and distinct values of the remaining indices, it was finally chosen for further research. The other k values were unclear due to small differences in index values and a lack of consensus onindex votes. However, the following cases were selected: the 3rd, the 6th (20 groups), the 1st (50 groups), and the 2nd (100 groups).

Evaluation in the Euclidean distance case was particularly interesting due to the high values of both indices S and CH. For 5 group schemes, S was from the interval [0.84-0.85], whereas CH from [350-432]. Ultimately, the 6th value was taken as it had the highest values. For 20 group schemes, the Silhouette index was slightly lower [0.81,0.82], but for all schemes, its values were rather equal. For this reason, the 2nd result was selected as the best due to the optimal values of the remaining indices. Similar procedures and reasons for option were performed for 50 group schemes - the 1st partitions were denoted as the best. The last set of results was completely different: values of S were near 0, and DB was 2 times higher than the previous ones. Calinski-Harabasz index was quite high - about 130 - however its values were equal. For this reason, it was decided not to take any scheme for $k = 100$.

Changing a clustering distance metric to CityBlock generated similar results to the previous one: both indices Silhouette

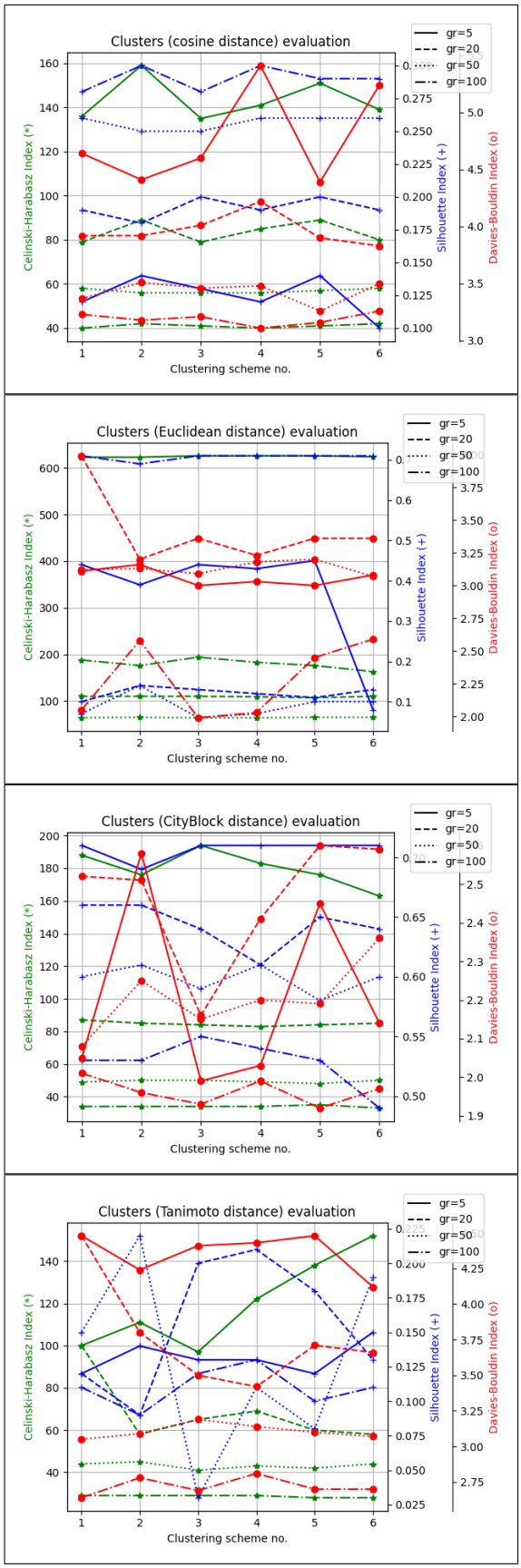

Fig. 2. Evaluation of clustering schemes on 100k dataset

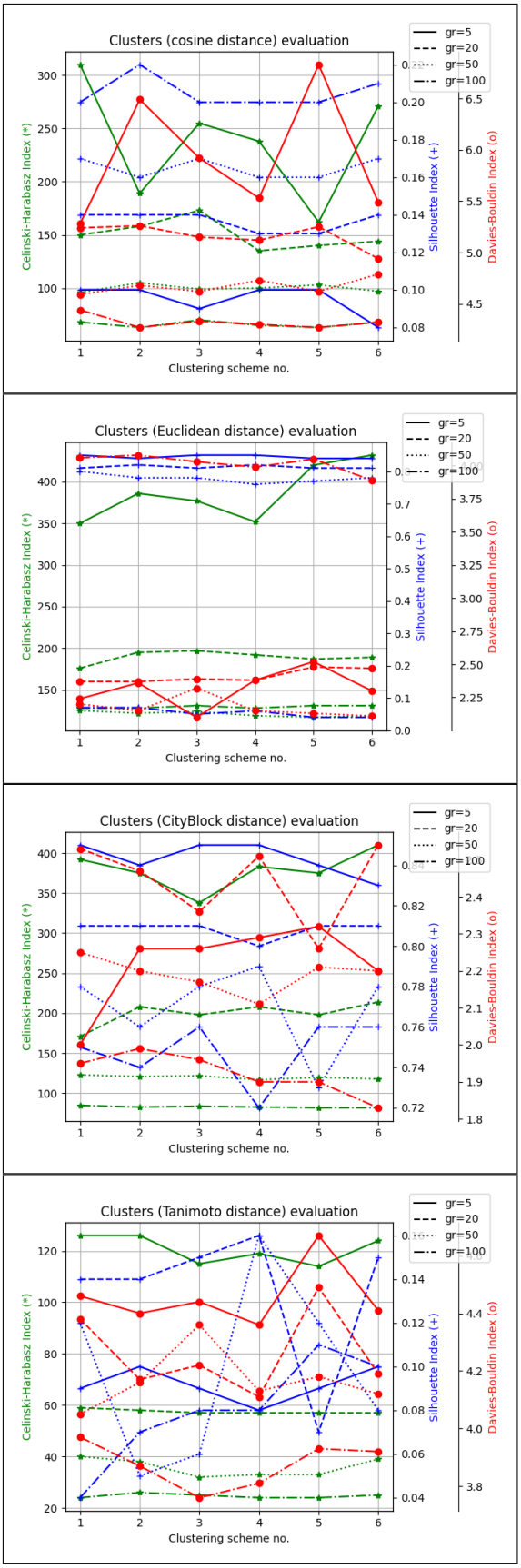

Fig. 3. Evaluation of clustering schemes on 1M dataset

and Calinski-Harabasz were high, and the Davies-Bouldin index was low. This advantage was present in all schemes for every value of k. Finally, the following clusterings were selected: the 1st (5 groups), the 3rd (20 groups), the 6th (50 and 100 groups) due to optimal values of both indices, DB and CH; S was equally distributed in the compared results.

In the last case, with the Tanimoto distance during clustering, was not as good as the previous cases: DB was rather high, S was near 0. Only the Calinski-Harabasz index was a bit valuable. Hence, only this one was taken into consideration and the following schemes were selected: the 4th (5 groups), the 4th (20 groups), the 1st (50 groups), and the 2nd, the 6th (100 groups).

B. Evaluation of Recommendations

The best clustering schemes were forwarded to $M-CCF$ recommender system, which was used to estimate missing ratings in the testing part of the datasets and then the calculated values were examined in contrast to the original ones in order to determine a difference in precision. Attention was paid, despite the accuracy, to the completeness of the recommendation lists generated by the systems

During this phase, the results obtained with the Multi-Clustering Collaborative Filtering algorithm (M-CCF) were compared against two baseline recommenders: the classical item-item memory-based approach (IBCF) and a system that utilized a single-clustering (SCCF) for neighborhood modeling.

The evaluation focused on assessing the accuracy and completeness of the recommendation lists generated by each system. Evaluation criteria were related to the following standard main metrics:

- Root Mean Squared Error ($RMSE$) described by (4) a baseline way to measure the error in model evaluation studies. It is a square root of an arithmetic mean of the squares of the predictions between the model and the observations (represented respectively, as $r_{real}(x_{ij})$ and $r_{est}(x_{ij})$ for user x_i and a particular item j). The rating r_{real} is a real user's rating, which is an integer number from interval [2,3,4,5]. The lower value of $RMSE$ refers to a better prediction ability.

$$RMSE = \frac{\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{k}\left(r_{real}(x_{ij}) - r_{est}(x_{ij})\right)^2}}{n \cdot k}$$
$$r_{real} \in [2,3,4,5],\ r_{est} \in \mathbb{R}_+ \tag{4}$$

- Coefficient of Determination (R^2) described by (5) [42] is an indicator that allows assessment of prediction using a simple linear regression. It measures a reproduction quality of real ratings by their estimated values. This is the value to be maximized towards 1 - the higher value of R^2 means the more the cloud of points narrows around the regression line.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\sum_{j=1}^{k}\left(r_{real}(x_{ij}) - r_{est}(x_{ij})\right)^2}{\sum_{i=1}^{n}\sum_{j=1}^{k}\left(r_{real}(x_{ij}) - \overline{r}(x_i)\right)^2}$$
$$r_{real} \in [2,3,4,5],\ r_{est} \in \mathbb{R}_+ \quad (5)$$
$$\overline{r}(x_i) = \frac{1}{k}\sum_{j=1}^{k} r_{real}(x_{ij}),\ r_{real} \in [2,3,4,5]$$

- $Coverage$ described by (6) measures the system's responsiveness to a required length of a recommendation list. It is a portion of generated predictions to the needed length. If a system covered all positions in the recommendation list, its $Coverage$ is 100%. During the evaluation process, there were cases in which the estimation of ratings was not possible. It often occurs in both $SCCF$ and $M-CCF$ when the item for which the calculations are performed, is not present in the same cluster to which the already rated items belong. In every experiment, it was assumed that both $RMSE$ and R^2 are significant if the value of $Coverage$ is greater than 90%.

$$Coverage = \frac{\sum_{i=1}^{N}\sum_{j=1}^{k} r_{est}(x_{ij})}{N} \cdot 100\%,\ r_{est} \in \mathbb{R}_+ \quad (6)$$

The symbols in the equations, as well as the method of calculation, are characterised in detail below. In all equations $\mathbb{R}$ stands for the set of positive real numbers, n is a number of users taken for evaluation, k is a number of ratings to be estimated and a number of required recommendations is denoted as N.

The performance of the compared approaches was evaluated in the following way. Before the clustering step, the whole input dataset was split into two parts: training and testing. The testing part's size was about 10% of the original data - detailed information is presented in Table I. The two last columns contain ratings\user and ratings\item ratio, respectively. In this step, the evaluation process remains consistent across all the experiments presented in the paper. The same test data is used throughout the evaluation, ensuring that the comparison between different recommender systems is more objective and fair. During the evaluation, the ratings from the test data are temporarily removed, simulating a real-world scenario where the recommender systems need to predict these missing values.

TABLE I
DESCRIPTION OF THE DATASETS USED IN THE EXPERIMENTS.

Dataset		No. of ratings	No. of users	No. of items	r\u ratio	r\i ratio
training	small - 100k	99 835	549	11 024	181.85	9.06
	big - 1M	991 116	5 430	16 384	182.52	60.49
test	small - 100k	1 176	110	969	10.69	1.21
	big - 1M	8 962	1088	3840	8.24	2.33

In the evaluation process, the values of ratings from the test part were removed and estimated by the recommender systems. Although the test set itself remained constant during the experiments, the values to remove and estimate were selected randomly every time. The difference between the original and the calculated value was taken for $RMSE$ calculation.

The purpose was to establish whether clustering evaluation indices are helpful in identifying the most valuable clusters, in order to increase the precision of recommendation lists, furthermore reducing the consumption of memory in $M-CCF$ algorithm.

This part of the experiments started from systems' evaluation on the small dataset, which contained 10 times fewer users and ratings with a slight difference in items' number. Tables II and III report the results - $RMSE$ and R^2 respectively - for all the configurations, Cosine-based, LogLikelihood, Pearson correlation, both Euclidean and CityBlock distance-based and Tanimoto coefficient were used to calculate affinity between items. In both tables mentioned above, the column with Pearson correlation is missing due to the value of $Coverage$ being below 90% in every case.

To have a compact view of the obtained results, without reduction of the general concept to confirm, in the experiments only selected results are reported, which were generated by both $SCCF$ and $M-CCF$ recommender systems with remaining all $IBCF$ outcomes presented. The configurations chosen for the experiments were carefully selected to represent both the best-performing settings and the most common or widely used configurations in terms of accuracy. The examples which were omitted had either similar or worse values and would not affect the general conclusions.

The tables contain an evaluation of $IBCF$ in the first row and selected configurations of $SCCF$ and $M-CCF$ systems that are formatted in the following way:

- SCCF-$distance$-x - x clusters generated by $k-means$ using cosine-based (cos) or Euclidean (eu) distance,
- M-CCF-$distance$-x-[y]-[z] - clusters generated by $k-means$ with $k = x, y, z$ (y and z are optional) in 6 runs, using one of the following distance measures: cosine-based (cos), Euclidean (eu), CityBlock (cb) or Tanimoto (tan),
- M-CCF-$distance$-x-[y]-[z]-s - clusters generated by $k-means$ on the conditions described above, however, the procedure of clustering schemes was applied.

In the case of $IBCF$ algorithm the results are not very good: $RMSE$ ranges from 0.91 to 0.94, R^2 varies from 0.61 to 0.76 with quite high $Coverage$ from 95% to 99%. $SCCF$ method obtained better values in some configurations: for data split into 5 groups for both clustering metrics: Euclidean and cosine-based: $RMSE$ ranges from 0.88 to 0.93 with $Coverage$ comparable to the previous results. In the case of the $SCCF$ method, rather than using just one configuration, a range of values is presented.

It refers to the characteristic of $k-means$ clustering algorithm, which often generates different results even if the values of its input parameters remain the same. Hence, it was launched 6 times and every particular clustering scheme

TABLE II
RMSE AND COVERAGE OF THE ALGORITHMS ON 100K DATASET. THE BEST VALUES ARE IN BOLD.

Algorithm	Similarity Measure				
	Cosine-based	LogLikelihood	Euclidean	CityBlock	Tanimoto
IBCF	0.94 (95%)	0.94(95%)	**0.92(95%)**	0.94 (99%)	**0.91 (95%)**
SCCF-cos-5	0.93-0.95 (95%)	0.93-0.94(95%)	**0.91-0.93(95%)**	0.92-0.94 (95%)	**0.91-0.92 (95%)**
SCCF-cos-20	0.94-1.00 (93%)	0.93-0.99(93%)	0.93-0.98(93%)	0.93-0.99 (93%)	0.92-0.98 (93%)
SCCF-cos-50	0.98-1.00 (91%)	0.98-1.00(95%)	0.97-0.99(97%)	0.97-0.99 (98%)	0.96-0.99 (98%)
SCCF-cos-100	-	0.96-1.03(94%)	0.95-1.02(97%)	0.96-1.02 (97%)	0.95-1.02 (98%)
SCCF-eu-5	**0.90-0.91 (95%)**	**0.90-0.91(97%)**	**0.89-0.90(98%)**	**0.90-0.91 (99%)**	**0.88-0.89 (99%)**
SCCF-eu-20	1.00-1.08 (93%)	0.99-1.08(96%)	0.99-1.08(98%)	1.00-1.10 (98%)	0.99-1.02 (98%)
SCCF-eu-50	-	1.00-1.03(95%)	1.02-1.03(97%)	1.03-1.05 (98%)	1.00-1.02 (98%)
SCCF-eu-100	-	0.99-1.00(94%)	0.99-1.02(96%)	1.02-1.03 (97%)	0.99-1.02 (97%)
M-CCF-eu-5	**0.91 (95%)**	**0.91 (95%)**	**0.88 (95%)**	**0.91 (95%)**	**0.90 (95%)**
M-CCF-eu-5-20	0.94 (93%)	0.98 (92%)	**0.92 (93%)**	1.01 (94%)	0.99 (92%)
M-CCF-eu-5-20-s	0.94 (93%)	0.98 (92%)	0.93 (93%)	1.01 (94%)	0.98 (92%)
M-CCF-cos-5-20-50	0.96 (91%)	-	0.94 (90%)	-	-
M-CCF-cos-5-20-50-s	0.93 (92%)	0.99 (90%)	**0.91 (91%)**	1.00 (91%)	0.98 (90%)
M-CCF-cb-5-20-50	0.95 (90%)	0.95 (90%)	0.93 (90%)	0.96 (90%)	0.93 (90%)
M-CCF-cb-5-20-50-s	0.94 (93%)	0.94 (93%)	**0.92 (93%)**	0.95 (93%)	0.93 (93%)
M-CCF-tan-5-20-50	0.97 (90%)	-	-	-	-
M-CCF-tan-5-20-50-s	0.96 (91%)	0.96 (90%)	0.93 (91%)	0.96 (90%)	0.95 (90%)

TABLE III
R2 AND COVERAGE OF THE ALGORITHMS ON 100K DATASET. THE BEST VALUES ARE IN BOLD.

Algorithm	Similarity Measure				
	Cosine-based	LogLikelihood	Euclidean	CityBlock	Tanimoto
IBCF	**0.76 (95%)**	0.73(95%)	0.69(95%)	0.61 (99%)	0.69 (95%)
SCCF-cos-5	**0.74-0.77 (95%)**	**0.75-0.77(95%)**	0.73-0.74(94%)	0.72-0.76 (95%)	0.73-0.74 (95%)
SCCF-cos-20	0.70-0.76 (93%)	0.70-0.77(93%)	0.70-0.75(93%)	0.67-0.76 (93%)	0.67-0.74 (93%)
SCCF-cos-50	0.72-0.76 (91%)	0.71-0.75(91%)	0.72-0.76(91%)	0.71-0.77 (91%)	0.68-0.75 (90%)
SCCF-cos-100	-	-	-	-	-
SCCF-eu-5	0.68-0.71 (95%)	0.69-0.73(95%)	0.67-0.70(95%)	0.67-0.70 (95%)	0.70-0.71 (95%)
SCCF-eu-20	0.72-0.73 (93%)	0.72-0.73(93%)	0.72-0.73(93%)	0.72-0.73 (94%)	0.72-0.73 (93%)
SCCF-eu-50	-	-	-	0.71-0.72 (92%)	-
SCCF-eu-100	-	-	-	-	-
M-CCF-eu-5	0.70 (95%)	0.74 (95%)	0.67 (95%)	0.67 (95%)	0.67 (95%)
M-CCF-eu-5-20	0.68 (93%)	0.71 (92%)	0.68 (93%)	0.70 (94%)	0.70 (92%)
M-CCF-eu-5-20-s	0.74 (94%)	0.74 (93%)	0.68 (93%)	0.74 (94%)	0.74 (93%)
M-CCF-cos-5-20-50	0.73 (91%)	-	0.71 (90%)	-	-
M-CCF-cos-5-20-50-s	0.74 (92%)	0.71 (90%)	0.73 (91%)	0.72 (91%)	0.71 (90%)
M-CCF-cb-5-20-50	**0.75 (90%)**	**0.77 (90%)**	0.71 (90%)	**0.77 (90%)**	**0.75 (90%)**
M-CCF-cb-5-20-50-s	**0.78 (93%)**	**0.77 (93%)**	0.72 (93%)	**0.76 (93%)**	**0.75 (93%)**
M-CCF-tan-5-20-50	**0.75 (90%)**	-	-	-	-
M-CCF-tan-5-20-50-s	**0.75 (91%)**	**0.75 (90%)**	0.73 (91%)	**0.75 (90%)**	**0.75 (90%)**

was evaluated individually and for this reason, the results are presented in a range. It also points out that the scheme selected for the recommendation process is not guaranteed optimal.

If the $Coverage$ was below 90% the result was not displayed in the tables - instead, there is a mark '-'. In the case of $M-CCF$ algorithm, the outcomes are comparable or frequently better. For instance, the configuration $M-CCF-eu-5$ and Euclidean-based similarity obtained $RMSE$=0.88 and the configuration $M-CCF-cos-5-20-50-s$ and the same similarity obtained $RMSE$=0.91. The best values of R^2 ($R^2=0.78$) measure were for the configuration $M-CCF-cb-5-20-50-s$. It must be admitted that the values of $Coverage$, although over 90%, are slightly lower than in the previous cases. The most important issue is that this experiment shows that the cluster schemes selection in terms of compactness and separability, for $M-CCF$ algorithm contributed mostly towards the performance of the model - usually, the values of both metrics $RMSE$ and R^2 as well as

$Coverage$ were improved. As an example, the configuration $M-CCF-cos-5-20-50-s$ can be presented, which $RMSE$ value decline was 0.03, whereas R^2 increased by 0.02 with simultaneous progress in $Coverage$.

The following experiment aimed to evaluate the systems on the big dataset, which contained 1 million ratings. Tables IV and V report the results - $RMSE$ and R^2 respectively. Similarity and clustering distance measures were the same as in the previous tests. Their configurations with schemes and a number of clusters were the same as well.

The results obtained by $IBCF$ algorithm were comparable to the previous outcomes on the small dataset. Only a slight decline was observed in R^2 values. In the case of $SCCF$, an improvement is visible - the values of $RMSE$ are lower for nearly all similarity measures examined, and the $Coverage$ is higher. The narrowing spread of values can be noted as well. A greater density of ratings in the data positively impacts the results.

The experiments and evaluation have revealed that the multi-clustering approach ($M-CCF$) shows greater progress compared to the single-clustering approach ($SCCF$) and the

TABLE IV
RMSE AND COVERAGE OF THE ALGORITHMS ON 1M DATASET. THE BEST VALUES ARE IN BOLD.

Algorithm	Similarity Measure					
	Cosine-based	LogLikelihood	Pearson corr.	Euclidean	CityBlock	Tanimoto
IBCF	0.95 (99%)	0.95(99%)	**0.92(98%)**	0.93(99%)	0.94 (99%)	**0.91 (99%)**
SCCF-cos-5	0.94(98%)	0.94(99%)	-	0.93(99%)	0.93(99%)	**0.91(99%)**
SCCF-cos-20	-	-	-	-	-	-
SCCF-cos-50	**0.90-0.91(96%)**	**0.90-0.91(98%)**	-	**0.90-0.91(99%)**	**0.89-0.91(99%)**	**0.89-0.90(99%)**
SCCF-cos-100	**0.91(94%)**	**0.91(97%)**	-	**0.90(98%)**	**0.90(99%)**	**0.90-0.91(99%)**
SCCF-eu-5	0.94-0.95 (98%)	0.94-0.95(99%)	-	**0.92-0.93(99%)**	0.93-0.94 (99%)	**0.91-0.92 (99%)**
SCCF-eu-20	0.94-0.95 (97%)	0.94-0.95(98%)	-	0.93-0.94(99%)	0.93-0.94 (99%)	**0.91-0.92 (99%)**
SCCF-eu-50	0.95(96%)	0.95(96%)	-	0.93-0.94(99%)	0.94(99%)	0.92(99%)
SCCF-eu-100	**0.91-0.92(93%)**	**0.91-0.92(96%)**	-	**0.90-0.91(98%)**	**0.91(98%)**	**0.90(98%)**
M-CCF-eu-5-20-50	0.95 (93%)	0.95 (94%)	**0.92(92%)**	0.93 (93%)	0.95 (94%)	**0.92 (93%)**
M-CCF-eu-5-20-50-s	0.94 (96%)	0.94 (96%)	**0.91(95%)**	0.93 (96%)	0.94 (96%)	**0.91 (96%)**
M-CCF-cos-5-20-50-100	**0.87 (92%)**	**0.90(91%)**	**0.90(91%)**	**0.86 (92%)**	**0.90(91%)**	**0.89 (91%)**
M-CCF-cos-5-20-50-100-s	**0.88 (94%)**	**0.90 (93%)**	**0.90(92%)**	**0.87 (94%)**	**0.90 (93%)**	**0.89 (93%)**
M-CCF-cb-5-20-50	0.95 (93%)	0.95 (93%)	0.93(92%)	0.93 (93%)	0.95 (93%)	**0.92 (93%)**
M-CCF-cb-5-20-50-s	0.94 (95%)	0.95 (95%)	0.92(93%)	0.93 (95%)	0.94 (95%)	**0.92 (95%)**
M-CCF-tan-5-20-50-100	0.93 (96%)	0.94 (96%)	**0.90(95%)**	**0.92 (96%)**	0.93 (96%)	**0.91 (96%)**
M-CCF-tan-5-20-50-100-s	0.93 (97%)	0.94 (96%)	**0.90(95%)**	**0.92 (96%)**	0.93 (96%)	**0.91 (96%)**

TABLE V
R2 AND COVERAGE OF THE ALGORITHMS ON 1M DATASET. THE BEST VALUES ARE IN BOLD.

Algorithm	Similarity Measure					
	Cosine-based	LogLikelihood	Pearson corr.	Euclidean	CityBlock	Tanimoto
IBCF	0.66(99%)	**0.68(99%)**	0.63(98%)	0.59(99%)	0.64 (99%)	0.64 (99%)
SCCF-cos-5	**0.68(98%)**	**0.69(98%)**	0.63(98%)	0.65-0.66(98%)	0.65-0.67(98%)	0.66(98%)
SCCF-cos-20	-	-	-	-	-	-
SCCF-cos-50	0.66(96%)	0.66(96%)	0.64(95%)	0.65-0.66(96%)	0.66(96%)	0.65(99%)
SCCF-cos-100	0.64-0.65(94%)	0.64-0.65(94%)	0.62-0.64(93%)	0.64-0.65(94%)	0.63-0.64(94%)	0.63-0.64(95%)
SCCF-eu-5	**0.67-0.68 (98%)**	0.67(98%)	0.63-0.64(98%)	0.63-0.64(98%)	0.66 (98%)	0.63(98%)
SCCF-eu-20	**0.68-0.69(97%)**	0.67-0.68(97%)	0.63-0.65(97%)	0.66-0.67(97%)	0.67 (97%)	0.65(97%)
SCCF-eu-50	**0.68-0.69(96%)**	**0.68-0.69(96%)**	0.64-0.65(95%)	0.67-0.68(96%)	0.67(96%)	0.65(96%)
SCCF-eu-100	0.64(93%)	0.64-0.65(93%)	0.64-0.65(91%)	0.64-0.65(93%)	0.64-0.65(93%)	0.63-0.64(93%)
M-CCF-eu-5-20-50	**0.67 (93%)**	**0.67 (94%)**	0.62(92%)	0.65 (93%)	0.65 (94%)	0.63 (93%)
M-CCF-eu-5-20-50-s	**0.69 (96%)**	**0.69 (96%)**	0.64(95%)	**0.68 (96%)**	**0.68 (96%)**	0.65 (96%)
M-CCF-cos-5-20-50-100	0.63 (92%)	0.62(91%)	0.62(91%)	0.61 (92%)	0.62(91%)	0.61 (91%)
M-CCF-cos-5-20-50-100-s	0.64 (94%)	0.66 (93%)	0.63(92%)	0.63 (94%)	0.65 (93%)	0.65 (93%)
M-CCF-cb-5-20-50	**0.68 (93%)**	**0.68 (93%)**	0.63(92%)	0.66 (93%)	**0.67 (93%)**	0.65 (93%)
M-CCF-cb-5-20-50-s	**0.67 (95%)**	**0.67 (95%)**	0.63(93%)	0.65 (95%)	0.66 (95%)	0.64 (95%)
M-CCF-tan-5-20-50-100	**0.68 (96%)**	**0.68 (96%)**	0.62(95%)	0.65 (96%)	**0.67 (96%)**	0.64 (96%)
M-CCF-tan-5-20-50-100-s	**0.68 (96%)**	**0.69 (96%)**	0.62(95%)	0.65 (96%)	**0.67 (96%)**	0.64 (96%)

classical item-item memory-based approach ($IBCF$). The values of $RMSE$ are considerably lower than in the experiments on the small dataset. The lowest value is 0.87 (for both similarity measures cosine-based and Euclidean), and it is the best result in all experiments described in this paper. Denser data were also beneficial, with $Coverage$ values which are higher than analogous outcomes obtained on the small dataset.

Comparison between the two versions of $M-CCF$ algorithm: without and with the selection of clustering schemes, shows that this step also improves the system's performance, however, the progress is not as great as was observed for $100k$ dataset. It can result from the fact that the clusters were evaluated well by the clustering indices, but any significant differences in their values were not observed.

Taking into consideration all the experiments presented in this article, it can be observed that the reduction in a number of clustering schemes did not negatively affect $M-CCF$ method - its performance was not declined, or it declined slightly. Furthermore, in numerous cases, the cluster selection procedure not only proved beneficial to the $M-CCF$ method without cluster selection but also improved the performance of both the $IBCF$ and $SCCF$ approaches. The recommendations generated by the selected clusters were more accurate and comprehensive compared to the recommendations provided by the unselected clusters.

VI. Conclusions

In this paper, a recommender system that is based on multi-clustering to model the neighbourhood of a target user, with internal indices-based clustering scheme selection was presented. The concept of $M-CCF$ algorithm is to store multiple clustering schemes on its input and dynamically matching every item that takes part in the recommendation generation process with the most appropriate cluster. As it advances in accuracy, it faces substantial challenges around memory consumption and time efficiency compared to recommender systems in which the neighbourhood of objects is identified by single-clustering schemes.

Clustering algorithms are not generally deterministic and thus may generate different clustering schemes even for the same value of an input parameter. Similarly, the results obtained by $k-means$ can be either similar or different from each other. Similar partitionings are redundant and do not contribute to the recommendation phase. The internal indices evaluate clustering schemes in terms of compactness and separability that allows making the selection of clustering schemes to forward for $M-CCF$ input.

An exclusive set of partitions definitely benefits memory usage by $M-CCF$ algorithm and, moreover, often its performance - $RMSE$ and $Coverage$. The results of the executed experiments confirmed that the performance of $M-CCF$ algorithm is often better when it works on a reduced set of input clusters. It was especially noticeable when the size of data was great (a dataset with 1 million ratings) - $RMSE$ was lower and $Coverage$ higher. Additionally, the technique still becomes free from the negative impact on the precision provided by the selection of an inappropriate clustering scheme as it occurs in the case of recommender systems in which the neighbourhood of objects is identified by single-clustering schemes.

The upcoming experiments will aim to verify the effectiveness of the proposed approach on larger datasets, such as those containing 10 million ratings. This evaluation on more substantial datasets will assess the scalability and performance of the Multi-Clustering Collaborative Filtering (M-CCF) algorithm in handling real-world, extensive datasets, thereby ensuring its practical applicability.

Additionally, the impact of different types of clustering algorithms will be investigated on the overall performance of the recommender system. By comparing the results obtained using various clustering algorithms, the study will identify the most suitable clustering technique that enhances recommendation accuracy and completeness.

Moreover, instead of relying on the output of a single clustering algorithm, the experiments will explore using a mixture of clustering schemes as input for the recommender system. This approach aims to assess whether combining multiple clustering methods can further improve recommendation quality and provide more diverse and personalized recommendations for users.

The evaluation will not be limited to accuracy alone; other characteristics of $M-CCF$, such as diversity, serendipity, and novelty, will also be measured. These aspects of the recommendations are crucial in providing users with novel and unexpected suggestions, contributing to an enriched user experience and engagement.

Overall, these comprehensive experiments will provide a deeper understanding of $M-CCF$'s performance, scalability, and adaptability, paving the way for its practical implementation in real-world, large-scale recommender systems.

References

[1] C. C. Aggrawal, *Recommender Systems. The Textbook*. Yorktown Heights: Springer, 2016.

[2] D. Jannach, *Recommender Systems: an Introduction*. New York: Cambridge University Press, 2010.

[3] J. Haupt, "Last.fm: People-powered online radio," *Music Reference Services Quarterly*, vol. 12, pp. 23–24, 2009.

[4] Bennett, J. and Lanning, S. , "The Netflix prize," in *Proceedings of KDD Cup and Workshop*, 2007.

[5] Corbeil, J. and Florent, D., "Deploying a Cost-Effective and Production-Ready Deep News Recommender System in the Media Crisis Context," in *Proceedings of RecSys*, 2020.

[6] K. Chaudhari and T. Ankit, "A comprehensive survey on travel recommender systems," *Archives of Computational Methods in Engineering*, vol. 27, no. 5, pp. 1545–1571, 2020.

[7] O. C. Agbonifo and A. Motunrayo, "A development of an ontology-based personalised e-learning recommender system," *International Journal of Computer (IJC)*, vol. 38, no. 1, pp. 102–112, 2020.

[8] X. Gao, F. Feng, H. Huang, X. Mao, T. Lan, and Z. Chi, "Food recommendation with graph convolutional network," *Information Sciences*, vol. 584, pp. 170–183, 2022. [Online]. Available: doi:10.1016/j.ins.2021.10.040

[9] J. Díez, P. Pérez-Núñez, O. Luaces, B. Remeseiro, and A. Bahamonde, "Towards explainable personalized recommendations by learning from users' photos," *Information Sciences*, vol. 520, pp. 416–430, 2020. [Online]. Available: doi:10.1016/j.ins.2020.02.018

[10] F. Ricci, L. Rokach, and B. Shapira, "Recommender systems: Introduction and challenges," in *Recommender Systems Handbook*, 2015, pp. 1–34.

[11] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutierrez, "Recommender systems survey," *Knowledge-Based Systems*, vol. 46, pp. 109–132, 2013.
[12] M. Singh, "Scalability and sparsity issues in recommender datasets: a survey," in *Knowledge and Information Systems*, 2018, pp. 1–43.
[13] Jannach, D. and Ludewig, M., "When recurrent neural networks meet the neighborhood for session-based recommendation," in *Proceedings of the Eleventh ACM Conference on Recommender Systems (RecSys17)*, 2017, p. 306–310.
[14] V. Yadav1, R. Shukla, A. Tripathi, and A. Maurya, "A new approach for movie recommender system using k-means clustering and pca," *Journal of Scientific & Industrial Research*, vol. 80, pp. 159–165, 2021.
[15] L. Kaufman, *Finding Groups in Data: An Introduction to Cluster Analysis*. New York: John & Sons Wiley, 2009.
[16] J. Bailey, "Alternative clustering analysis: a review," in *Intelligent Decision Technologies: Data Clustering: Algorithms and Applications*. Boca Raton: Chapman and Hall/CRC, 2014, pp. 533–548.
[17] T. Li, M. Ogihara, and S. Ma, "On combining multiple clusterings: An overview and a new perspective," *Applied Intelligence*, vol. 33, no. 2, pp. 207–219, 2010.
[18] U. Kużelewska, "Effect of Dataset Size on Efficiency of Collaborative Filtering Recommender Systems with Multi-clustering as a Neighbourhood Identification Strategy," in *International Conference on Computational Science*. New York: Springer, 2020, pp. 342–354.
[19] ——, "Scheme Selection based on Clusters' Quality in Multi-Clustering M-CCF Recommender System," in *31st International Conference on Information Systems Development (ISD 2023)*, 2023.
[20] S. Latifi, N. Mauro, and D. Jannach, "Session-aware recommendation: A surprising quest for the state-of-the-art," *Information Sciences*, vol. 573, pp. 291–315, 2021. [Online]. Available: doi:10.1016/j.ins.2021.05.048
[21] R. Kuo, C. Chen, and S. Keng, "Application of hybrid metaheuristic with perturbation-based k-nearest neighbors algorithm and densest imputation to collaborative filtering in recommender systems," *Information Sciences*, vol. 575, pp. 90–115, 2021. [Online]. Available: doi:10.1016/j.ins.2021.06.026
[22] Y. Kilani, A. Otoom, A. Alsarhan, and M. Almaayah, "A genetic agorithms-based hybrid recommender system of matrix factorization and neighborhood-based techniques," *Journal of Computational Science*, vol. 28, pp. 78–93, 2018. [Online]. Available: doi:10.1016/j.jocs.2018.08.007
[23] U. Kużelewska, "Dynamic Neighbourhood Identification Based on Multi-clustering in Collaborative Filtering Recommender Systems," in *International Conference on Dependability and Complex Systems*, 2020, pp. 410–419.
[24] Yaoy, S. and Yuy, G. and Wangy, X. and Wangy, J. and Domeniconiz, C. and Guox, M., "Discovering Multiple Co-Clusterings in Subspaces," in *Proceedings of the 2019 SIAM International Conference on Data Mining*, 2019, pp. 423–431.
[25] A. Bilge and H. Polat, "A scalable privacy-preserving recommendation scheme via bisecting k-means clustering," *Information Process Management*, vol. 49, no. 4, pp. 912–927, 2013.
[26] M. Farahani, J. Torkestani, and M. Rahmani, "Adaptive personalized recommender system using learning automata and items clustering," *Information Systems*, vol. 106, p. 101978, 2022. [Online]. Available: doi:10.1016/j.is.2021.101978
[27] Rashid, M. and Shyong, K. L. and Karypis, G. and Riedl, J., "ClustKNN a Highly Scalable Hybrid Model - Memory-based CF Algorithm," in *Proceeding of WebKDD*, 2006.
[28] L. R. Divyaa and N. Pervin, "Towards generating scalable personalized recommendations: Integrating social trust, social bias, and geo-spatial clustering," *Decision Support Systems*, vol. 122, 2019.
[29] R. Logesh, V. Subramaniyaswamy, D. Malathi, N. Sivaramakrishnan, and V. Vijayakumar, "Enhancing recommendation stability of collaborative filtering recommender system through bio-inspired clustering ensemble method," *Neural Computing and Applications*, vol. 32, pp. 2141—-2164, 2020.
[30] S. Kant and T. Mahara, "Nearest biclusters collaborative filtering framework with fusion," *Journal of Computational Science*, vol. 25, pp. 204–212, 2018. [Online]. Available: doi:10.1016/j.jocs.2017.03.018
[31] F. de Aguiar Neto, A. da Costa, M. Manzato, and R. Campello, "Preprocessing approaches for collaborative filtering based on hierarchical clustering," *Information Sciences*, vol. 534, pp. 172–191, 2020. [Online]. Available: doi:10.1016/j.ins.2020.05.021
[32] S. Bansal and N. Baliyan, "Bi-mars: A bi-clustering based memetic algorithm for recommender systems," *Applied Soft Computing*, vol. 97, p. 106785, 2020. [Online]. Available: doi:10.1016/j.asoc.2020.106785
[33] P. Fränti and S. Sieranoja, "How much can k-means be improved by using better initialization and repeats?" *Pattern Recognition*, vol. 93, pp. 95–112, 2019.
[34] A. Strehl and J. Ghosh, "Cluster ensembles – a knowledge reuse framework for combining multiple partitions," *Journal of Machine Learning Research*, vol. 3, pp. 583–617, 2002.
[35] L. Bai, Y. Liang, and F. Cao, "A multiple k-means clustering ensemble algorithm to find nonlinearly separable clusters," *Information Fusion*, vol. 61, pp. 36–47, 2020. [Online]. Available: doi:10.1016/j.inffus.2020.03.009
[36] S. Zahra, M. A. Ghazanfar, A. Khalid, M. A. Azam, U. Naeem, and A. Prugel-Bennett, "Novel centroid selection approaches for kmeans-clustering based recommender systems," *Information Sciences*, vol. 320, pp. 156–189, 2015. [Online]. Available: doi:10.1016/j.ins.2015.03.062
[37] P. J. Rousseeuw, "Silhouettes a graphical aid to the interpretation and validation of cluster analysis," *Computational and Applied Mathematics*, vol. 20, p. 53–65, 1987.
[38] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *PAMI-IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1, no. 2, pp. 224–227, 1979.
[39] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics - Theory and Methods*, vol. 3, pp. 1–27, 1974.
[40] "Movielens dataset." [Online]. Available: https://grouplens.org/datasets/movielens/25m/
[41] F. Pedregosa, "Scikit-learn: Machine learning in python," *JMLR*, vol. 12, pp. 2825–2830, 2011.
[42] J. Miles, *R squared adjusted R squared*. New York: Wiley StatsRef: Statistics Reference Online, 2014. [Online]. Available: doi:10.1002/9781118445112.stat06627