Article

Community-Based Matrix Factorization (CBMF) Approach for Enhancing Quality of Recommendations

Srilatha Tokala , Murali Krishna Enduri , Jaya Lakshmi T , Hemlata Sharma *

Posted Date: 7 August 2023

doi: 10.20944/preprints202308.0518.v1

Keywords: Matrix factorization; Recommendation; Community detection; Parallel computation; RMSE

Article

Community-Based Matrix Factorization (CBMF) Approach for Enhancing Quality of Recommendations

Srilatha Tokala [1], Murali Krishna Enduri [1,*], T. Jaya Lakshmi [1] and Hemlata Sharma [2,*]

[1] Room No 203, Algorithms and Complexity Theory Lab, Department of Computer Science and Engineering, SRM University-AP, Amaravati, India; srilatha_tokala@srmap.edu.in (S.T.); jayalakshmi.t@srmap.edu.in (T.J.L.)
[2] Department of Computing, Sheffield Hallam University, UK
* Correspondence: muralikrishna.e@srmap.edu.in (M.K.E.); h.sharma@shu.ac.uk (H.S.)

Abstract: Matrix Factorization is a long established method employed for analyzing and extracting valuable insight recommendations from complex networks containing user ratings. The execution time and computational resources demanded by these algorithms pose limitations when confronted with large datasets. Community detection algorithms play a crucial role in identifying groups and communities within intricate networks. To overcome the challenge of extensive computing resources, we present a novel parallel computation framework utilizing community information available in the rating network. Our proposed approach named as Community-based Matrix Factorization(CBMF), parallelizes matrix factorization technique by dividing the network into communities using existing community detection algorithms. We prove that this parallel approach not only increases the quality of recommendations in connection with Root Mean Square Error (RMSE), but also yields substantial performance improvement. We empirically evaluate our idea on diverse datasets and present comprehensive experimental results. These results serve as empirical evidence of the effectiveness and performance gains offered by our parallel computation framework.

Keywords: matrix factorization; recommendation; community detection; parallel computation; RMSE

1. Introduction

To predict recommendations to the users based on past behavior and their preferences, a technique in recommender systems namely collaborative filtering is used [1]. The main objective is to suggest items or content to the users by considering their interactions or resemblances with other users. Due to the continuous growth in data availability and the interconnected nature of diverse systems, these techniques hold immense importance in revealing patterns, relationships, and significant insights [2].

Matrix Factorization (MF) is a technique utilized in collaborative filtering to break down a matrix of user-item ratings into lower-rank matrices capturing the latent factors, underlying the data [3,4]. The user-item rating matrix serves as a representation of user ratings assigned to different items [5]. The matrix is frequently sparse as users typically only rate a small subset of items. MF techniques strive to complete the missing entries in the matrix by decomposing it into lower-rank matrices, one capturing the user's underlying preferences and the other reflecting the item's latent characteristics [6]. The latent representations of users along with items can be used to estimate future ratings or calculate missing ratings once the matrix has been factorized.

In network analysis, community detection stands as a fundamental task, endeavoring to identify cohesive subsets of nodes, referred to as communities or modules within a network [7]. These communities in networks signify groups of nodes that display stronger interconnections amongst themselves compared to connections with nodes outside the community [8]. These communities provide insightful information on the structure, organization, and dynamics of complex systems [9,10]. The applications of community detection across various domains such as social network analysis, biological networks, online forums, and recommendation systems have garnered substantial attention [11]. Through the identification of communities, we gain insights into the underlying interaction patterns, discover influential groups, and develop a deeper comprehension of the network's functionality [12].

1.1. Applications

Matrix Factorization techniques have been widely applied across diverse domains, demonstrating their versatility and effectiveness. Some of the notable applications of the matrix factorization techniques include

1. **Natural Language Processing (NLP)**: Within the field of NLP, techniques for matrix factorization have been used in a variety of tasks in topic modeling, text classification, etc [13,14]. Through the decomposition of the document matrix, MF algorithms can reveal latent representations that effectively capture the underlying semantic structure of the textual data.
2. **Social Network Analysis**: MF techniques have been utilized in social network analysis to unveil communities of individuals sharing common interests or behaviors [15,16]. SVD++ facilitates the identification of friends, influencers, or interest-based communities by enhancing user experience and engagement on social media platforms. FANMF is used to uncover community structures and identify influential nodes within the network. FANMF can effectively detect the group of nodes by exposing hidden relationships and structures by factorizing them into non-negative adjacency matrices.
3. **E-commerce**: In the realm of e-commerce platforms, SVD++ is extensively utilized to deliver personalized product recommendations to users [17,18].By including implicit feedback, supplementary data, and user-item ratings, SVD++ can adeptly capture user preferences and item characteristics for accurate product recommendations [19].
4. **Streaming Services**: Streaming platforms including music or video services, SVD++ provide personalized content recommendations to the users. SVD++ significantly enhances the discovery and recommendation of relevant and captivating content. It ensures that the users are presented with content aligned with their individual tastes and preferences [20].
5. **Image Processing**: FANMF finds application in image processing tasks, where the image data is factored into non-negative matrices. By extraction, it improves image quality and facilitates the analysis of visual data. By decomposition, matrix factorization algorithms are capable of distinguishing noise from the underlying structure. It also completes missing parts, and extracts significant features for analysis and representation [21].

In essence, the applications of MF techniques have a broad scope of transforming the methods through which we analyze, comprehend, and leverage complex data [22]. The ongoing evolution of these techniques holds the potential to unlock fresh possibilities and propel advancements in numerous domains. It ultimately benefits individuals, organizations, and society at large.

2. Literature Review

In recent years, the MF method has garnered significant attention as a widely adopted and successful method for rating prediction in recommendation systems. The Netflix Prize competition, which was started in 2006, is one early piece of work noteworthy in relation to MF in recommender systems [23]. The fundamental matrix factorization model creates user and item latent feature matrices from the rating matrix, enhancing the accuracy of rating predictions through the understanding of possible connections between users and items.

As the information interconnection era has emerged, the basic MF model is no longer satisfied the demands of recommender systems. It thus leads to the emergence of numerous variants of this model. Excluding all the non-negative entries in latent features, a model Non-Negative Matrix Factorization (NMF) has been initiated by Paatero *et al.*, and Tapper *et al.* in 1994 that improves the accuracy of the model [24]. R.Salakhuntidnov *et al.*, and A.Mnih *et al.* in the year 2007 proposed Probabilistic Matrix Factorization (PMF) that utilizes probabilistic modeling to effectively capture uncertainties in user-item ratings, resulting in recommendations that are more reliable and precise [25]. The significance of the Singular Value Decomposition (SVD) technique is introduced by Eugenio Beltrami *et al.*, and Camille Jordan *et al.* in 1857 and came into existence from 2006 [26]. The method has the ability to perform robust MF, enabling the identification of hidden features and effective dimensionality reduction in data analysis.

By integrating explicit and implicit feedback to enhance recommendation accuracy and personalization, Yehuda Koren *et al.* in the year 2008 introduced Advanced Singular Value Decomposition (SVD++) method [27]. Through the incorporation of user-item ratings and implicit feedback, SVD++ boosts the performance of the recommender systems, enabling better capture of user preferences and more effective recommendation generation. In 2015, Xiaohua Shi *et al.* introduced a method namely Pairwisely Constrained Nonnegative Symmetric Matrix Factorization (PCSNMF) that incorporates the symmetric community structures found in undirected networks but also leverages pairwise constraints derived from ground-truth group information [28]. By doing so, we enhance the accuracy of community detection. In 2017, Deep Matrix Factorization (DMF) is a significant technique introduced by Xiangnan He *et al.* that emerges the deep learning with MF [29]. This method enables the discovery of intricate patterns and the extraction of complex features from large-scale datasets. Kipf *et al.*, and Welling *et al.* in the year 2018 combines the power of MF and graph convolutional neural network to capture both collaborative filtering patterns and graph structures in recommendation systems [30]. By incorporating the graph information, the recommendation accuracy is improved by leveraging the connectivity and relationships among users and items. In 2019, Factorized Asymmetric Non-Negative Matrix Factorization (FANMF) was introduced by Tosyali *et al.* by considering the asymmetric relationships between users and items [31]. It leads to enhanced recommendation quality, a deeper understanding of user preferences, and ultimately providing personalized user experiences in various data analysis tasks.

M. Girvan *et al.* and M. E. J. Newman *et al.* in 2002 proposed an algorithm namely Girvan-Neuman that is very significant because it uses edge betweenness centrality to iteratively remove edges and identify communities in a network [32]. In 2007 Raghavan *et al.*, Albert *et al.*, and Kumara *et al.* initiated a label propagation algorithm that is a straightforward and efficient approach, updating node labels iteratively based on the majority of their neighbors to effectively detect communities in a network [33]. Vincent D. Blondel *et al.*, Jean-Loup Guillaume *et al.*, Renaud Lambiotte *et al.*, and Etienne Lefebvre *et al.* in 2008 proposed an algorithm Louvain that has the ability to efficiently optimize modularity, making it highly effective in detecting communities in large-scale networks while maintaining computational speed [34]. In 2011 by Pascal Pons *et al.*and Matthieu Latapy *et al.* introduced the Walktrap algorithm that is notable for its use of random walks to assess node similarities, providing an effective technique for locating communities in large networks [35]. The Leiden algorithm was bought up in 2019 by V.A.Traag *et al.*, L.Waltman *et al.*, and N. J. van Eck *et al.* to ensure there are connected communities, and iterative application results in a partition when local best assignments are made to all community subsets.

3. Methodology

In this work, we come up with a parallel framework for Matrix Factorization. This section first provides an elaborate exposition elucidating the various Matrix Factorization methods. Then discusses the community detection algorithms.

3.1. Matrix Factorization (MF)

In the early 2000*s*, the concept of matrix factorization emerged from the field of linear algebra. It can be used in diverse domains like recommender systems and data analysis. Simon Funk's work during the Netflix Prize competition in 2006 played a significant role in popularizing the use of matrix factorization algorithms for recommender systems [23].

In the MF method, we compute the latent features based on user-item interactions of which a matrix serves as a representation [36]. Ratings are used to describe the interactions between users and items in the matrix, which is created by placing people on one axis and items on the other [37]. The representation of the matrix is shown below.

$$\begin{bmatrix} r_{11} & .. & r_{1b} \\ . & .. & . \\ . & .. & . \\ . & .. & . \\ r_{m1} & .. & r_{mn} \end{bmatrix} = \begin{bmatrix} m_{11} & .. & m_{1k} \\ . & .. & . \\ . & .. & . \\ . & .. & . \\ m_{a1} & .. & m_{ak} \end{bmatrix} \begin{bmatrix} n_{11} & .. & n_{1b} \\ . & .. & . \\ . & .. & . \\ . & .. & . \\ n_{k1} & .. & n_{kb} \end{bmatrix}$$

$a \times b$ $a \times k$ $k \times b$ a stands for users, b for items, and k for the number of features that were extracted. M is a matrix representing the latent features of the users, and N denotes the latent features of the items. The relation between R and $\widetilde{R}$ is shown in Equation 1.

$$R \approx XY^T \tag{1}$$
$$\widetilde{R} = XY^T$$

The user and item rating vectors are combined to create the original rating R as shown in Equation 2.

$$r_{ab} \approx m_a n_b^T \tag{2}$$

The values of M and N are randomly distributed from 0 to 1. We compute the empty ratings in the matrix by minimization of the squared error as shown in Equation 3.

$$min \sum_{a,b} (r_{ab} - m_a n_b^T)^2 \tag{3}$$

In order to avoid overfitting the squared error, the regularization term α is added as shown in Equation 4.

$$min \sum_{a,b} (r_{ab} - m_a n_b^T)^2 + \alpha(||m_a||^2 + ||n_b||^2) \tag{4}$$

where $||.||$ is the frobenius norm. The approximation of this value is calculated using stochastic gradient descent or alternating least squares. For each rating in the training data, the prediction error is calculated using the stochastic gradient descent method as displayed in Equation 5.

$$e_{ab} = r_{ab} - m_a n_b^T \tag{5}$$

Then update the values of n_b and m_a with β learning rate and α regularization constant is added as shown in Equation 6.

$$\begin{aligned} n_b &\longleftarrow n_b + \beta(e_{ab} m_a - \alpha n_b) \\ m_a &\longleftarrow m_a + \beta(e_{ab} n_b - \alpha m_a) \end{aligned} \tag{6}$$

The dot product of the prediction matrix that is obtained and the latent feature matrices m_a and n_b as shown in Equation 7.

$$\widetilde{r} = m_a . n_b \tag{7}$$

After computing the predicted rating matrix, the difference between the cells of R and $\widetilde{R}$ as shown in Equation 8 is computed as Root Mean Square (RMSE).

$$RMSE = \sqrt{\frac{1}{N}\sum(r_{ab} - \widetilde{r}_{ab})^2} \tag{8}$$

where r_{ab} is the original rating, $\widetilde{r}_{ab}$ is the predicted rating, and N is the total number of predictions.

3.2. SVD++

It is an extension of the singular value decomposition method introduced by Yehuda Koren in 2008 [27]. He proposed this method as a part of his work improvement in the Netflix recommendation algorithm.

In the SVD++ method, implicit feedback is added to the user's latent feature vector [38,39]. The implicit feedback PQ is added and is calculated as follows:

P **Calculation:**

In the P matrix, the values will be either 1 or 0. The value of $P = [p_{ab}]\ \forall(a, b)$ is 1 if r_{ab} is observed or else 0. Every non-zero entry in the P matrix is written as $\frac{1}{\sqrt{|I_i|}}$ which is a $u \times v$ matrix.

Q **Calculation:**

$Q = [q_{ab}]\ \forall(a, b)$ which is similar to the item latent feature vector of order $v \times k$.

Calculate the dot product of P and Q matrices and add it to the user latent feature matrix i.e.,$(M + PQ)$. Then the deviation between the original rating R i.e., $(m_a n_b)$ and the predicted rating matrix $\widetilde{R}$ i.e., $((m + pq)_a n_b)$ is computed as the RMSE value as shown in Eq. (8).

3.3. Factorized Asymmetric Non-Negative Matrix Factorization (FANMF)

In 2019, the FANMF method is introduced by Tosyali [31]. The main purpose of the FANMF method is to handle non-negative data that has an asymmetric nature. Data in the real world is often asymmetry, where the rows and columns are related in this way are not symmetrical [40]. In the FANMF method, a rating matrix is decomposed into two non-negative latent feature matrices. FANMF is different from the traditional methods, as it accommodates the feature matrices of different dimensions that enable to capture the of data that are asymmetric in nature. It also adds a sparsity constraint along with the non-negativity constraint, that contains only a limited number of non-zero elements.

FANMF extends NMF to asymmetric scenarios where the user and item bias are considered in addition to the user-item interactions to make the model more accurate and improve the recommendation quality. User item bias includes the inherent inclinations of users towards specific items or the inherent attractiveness of items to users, independent of their previous interactions or behaviors.

The original rating matrix is represented as the product of user and item rating vectors as shown in Equation 9.

$$r_{ab} \approx m_a n_b^T \tag{9}$$

Update the values of M and N by using a multiplicative update algorithm that contains update values [41].

$$\begin{aligned} M &= M \cdot \times ((R \cdot / (M \times N + (R == 0))) \times N^T) \\ N &= N \cdot \times (M^T \times (R \cdot / (M \times N + (R == 0)))) \end{aligned}$$

where $M \cdot \times N$ is the dot product of M and N similarly $M \cdot / N$ is the dot division of M and N i.e., element wise division. $M \times N$ is the product of two matrices M and N. M^T is the transpose of the matrix M. The term $R == 0$ is included in the denominator to prevent division by zero.

Make the dot product of the updated latent feature matrices which is $\widetilde{R}$. The deviation between the original and the predicted rating matrices is computed as RMSE value as shown in Equation 8.

In this work, we introduce parallelism in matrix factorization with the help of community information. Various community detection algorithms are available in the literature to divide a complex network into communities. The Girvan-Newman is one of the first popular algorithms that uses edge betweenness centrality to detect communities in networks, achieved through iterative removal of edges with high betweenness. The shortcoming of the above-mentioned algorithms is while it excels at identifying overlapping communities, it faces the drawback of being computationally expensive, particularly for large graphs, due to repeated edge removals and recalculations of betweenness. Louvain algorithm addresses these shortcomings by adopting an efficient approach centered around optimizing modularity. Through iterative modularity optimization with node movements, it achieves fast and scalable community detection, effectively capturing communities even in large-scale networks. Louvain's superior optimization of modularity enables it to outperform Girvan-Newman in terms of speed and scalability, making it the preferred choice for real-world community detection tasks involving massive and complex networks. Therefore, we use the Louvain algorithm to divide the rating network into communities. Next section gives the details of Louvain.

3.4. Louvain Algorithm

The Louvain algorithm works effectively in covering large communities or groups that are densely interconnected in a network. It employs a modularity optimization process, iteratively refining the network's community structure to maximize modularity.

Louvain algorithm has a 6 step procedure to find the appropriate communities. Each node in the initial step is attached to its own community. Next, rearrange the nodes between the communities that iteratively optimize the modularity score which elevates the standards of the community structure. By iteratively considering each node, evaluate the modularity score by relocating the nodes to nearby communities. It is followed until there is no change in the modularity score. A new graph is constructed by combining the nodes in each community by connecting the nodes and edges of different communities. All these steps are repeated until there is no change in the modularity score is observed.

The Louvain algorithm has become popular because of its efficiency in detecting communities in the network by achieving a high modularity score, all while maintaining computational efficiency [9]. A higher modularity score signifies a more robust community structure, characterized by tightly interconnected nodes within communities while having fewer connections between communities [42,43]. This indicates the algorithm's successful identification of distinct and internally cohesive communities, demonstrating their meaningful organization within the network.

4. Proposed Method

In any type of MF method, the rating matrix is constructed by using users and items. The predicted rating matrix is obtained by using different update rules for the latent feature matrices in each kind of MF. The variation between the original and the predicted rating matrix is defined as the RMSE value. In order to run such huge rating matrices, the computational time and efficiency are very high. To reduce that, a combination of the MF method with the community detection algorithm "Louvain" is used.

The following procedure is carried out for using this approach as shown in Algorithm 1.

Algorithm 1: Community-based Matrix Factorization Algorithm (CBMF)

```
Input: $R$: Rating matrix of size $m \times n$, containing user-ratings. $m$ is the number of users and $n$
       is the number of items
Output: RMSE Value
begin
    Step 1: Construct a bipartite graph $BG$ from $R$. Step 2: Apply the Louvain algorithm to $BG$
      to generate $c$ communities. Let the community partition is denoted by $C$, where
      $C = BG_1 \bigcup BG_2 \bigcup \ldots \bigcup BG_c$
    for $i$ = 1 to $c$ do do
        Step 3: Extract rating matrix $R_i$ corresponding to $BG_i$ from $R$.
        Step 4: Apply the chosen MF method to the rating matrix $R_i$ and generate predicted
          rating matrix $\widetilde{R}_i$.
    end
    Step 5: Combine predicted rating matrices generated for each community into a single
      predicted rating matrix $\widetilde{R}$.
    Step 6: Calculate RMSE value from the original rating matrix $R$ and the predicted rating
      matrix $\widetilde{R}$.
    Return RMSE
end
```

A bipartite graph BG is constructed from users, and items in the dataset. The procedure followed for the construction of a bipartite graph is: Consider two sets of nodes, one for users and one for items. For each user node in one set, establish connections to every item node in the other set to represent the interactions or relationships between users and items in the dataset. If there exists a relation between the user and the item, there establishes an edge between the user and the item. For the bipartite graph that is constructed, the Louvain algorithm is applied and divided into communities. A rating matrix is formed by each community that is obtained from the graph. The size of the rating matrix is the same as the community size. For the rating matrices that are obtained, we apply MF (i.e., MF, SVD++, or FANMF) method, and by using the update rules we perform updation on the rating matrix, and the predicted rating matrices are formed. Combine all the predicted rating matrices into a single prediction rating matrix. RMSE value is calculated between the original rating matrix that is obtained before dividing into communities with the predicted rating matrix that is obtained. The flowchart of the overall approach is shown in Figure 1.

It is observed that we can get less RMSE value when dividing into communities by without division. At some times a constant RMSE value is maintained after a certain number of communities. The time taken for computing the RMSE value is also low due to parallel computation.

Hence, we can say that the computational time and efficiency are better as compared to only using the MF method. A combination of the MF method with community detection gives better results.

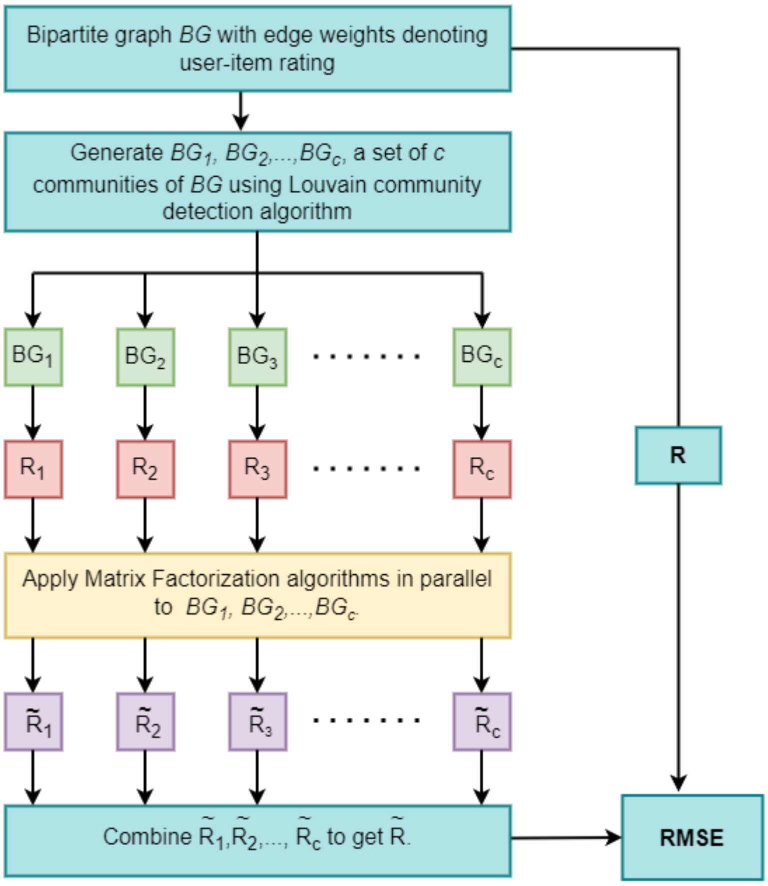
Bipartite graph BG with edge weights denoting user-item rating
Generate BG1, BG2,...,BGc, a set of c communities of BG using Louvain community detection algorithm
BG1
BG2
BG3
BGc
R1
R2
R3
Rc
R
Apply Matrix Factorization algorithms in parallel to BG1, BG2,...,BGc.
R̃1
R̃2
R̃3
R̃c
Combine R̃1,R̃2,..., R̃c to get R̃.
RMSE

Figure 1. Community-based Matrix Factorization (CBMF) framework.

5. Time Complexity

In computer science and algorithm analysis, the definition of time complexity plays a vital role. By quantifying time complexity, we can compare various methods for problem-solving and assess their performance considering input sizes or limitations. This comparison facilitates algorithm selection and leads us to discover more efficient solutions. Let us consider the time complexities of our different matrix factorization methods (MF, SVD++, and FANMF) in this section.

For the Matrix Factorization method, let $a \times b$ be a rating matrix of a users, b items, and k latent features. The iterations are same for every method and are taken as constant. The time complexity required for calculating the MF method is typically considered to be $\mathcal{O}(abk)$ [44]. The Louvain community detection method takes $\mathcal{O}(a \log a)$ time complexity, where a is the graph's number of nodes [45].

For a bipartite graph BG that contains users and items is taken as input for the louvain algorithm, the total of users and items will equal the number of nodes in the graph. Here, in this case, $(a+b)$ will be the nodes in the graph. Therefore, the time complexity of the louvain algorithm will be $\mathcal{O}((a+b)\log(a+b))$. A flow graph G is constructed from the bipartite graph and is divided into appropriate community structures by using the louvain algorithm (say $c_1, c_2, \cdots, c_c$). From each community, a small rating matrix will be obtained (say $R_1, R_2, \cdots, R_c$). Apply MF time complexity for each rating matrix and we get a time complexity of $\mathcal{O}(a_1 b_1 k_1)$, $\mathcal{O}(a_2 b_2 k_2)$,$\cdots$, $\mathcal{O}(a_c b_c k_c)$, consider the maximum of all these time complexities. Suppose R_p community rating matrix has the maximum value and the time complexity is $\mathcal{O}(a_p b_p k_p)$.

Hence, the overall time complexity for the MF method using the louvain community detection method is $\mathcal{O}((a+b)\log(a+b)) + \mathcal{O}(a_p b_p k_p)$.

Similarly for SVD++ and FANMF methods, the time complexity is $\mathcal{O}(abk)$. Therefore the total time complexity that is required for the calculation of SVD++, and FANMF methods will be $\mathcal{O}((a+b)\log(a+b)) + \mathcal{O}(a_p b_p k_p)$.

6. Dataset Statistics

To process our analysis, we have taken different datasets from different domains. Datasets namely Movie Lens-100K, Film Trust, Jester, Good Books, Wikilens, and Cell Phone Recommendation are taken. The dataset statistics of all these datasets are shown in Table 1.

Table 1. Dataset statistics for six different datasets namely Movie Lens-100K, Film Trust, Jester, Wikilens, Good Books, and Cell Phone Recommendation datasets.

Dataset	Number of Users	Number of Items	Number of Ratings	Rating Range	Average Rating	Sparsity
Movie Lens-100K	943	1682	100000	1-5	3.529	0.937
Film Trust	1508	2071	35494	0.5-4	3.002	0.988
Jester	31958	140	1048575	-10 - +10	0.955	0.839
Wikilens	326	5111	26936	0.5-5	3.468	0.983
Good Books	13123	7774	1048575	1-5	3.806	0.989
Cell Phone Recommendation	99	33	990	1-10	6.689	0.708

All of these dataset's distribution plots are displayed in Figure 2. The plots are drawn by taking the ratings, the X-axis represents users, while the Y-axis represents the count of each rating provided by those users. In the dataset movie lens-100K, the users are the people and the items are the movies in the dataset. The rating distribution is from 1 to 5 where the people have rated movies. It is observed in the plot that 34174 of the users had given a higher rating of four for the movies and a low rating of one is given by 6110 the users out of 100000 ratings. For the dataset film trust, the users are the people and the items are the films having a rating distribution from 0.5 to 4 where the people have rated films. It is observed in the plot that 9170 users had given a rating of five for films and a low rating of 0.5 is given by the users 1060 out of 35494 ratings. In the dataset jester, the users are the people and the items are jokes having a rating distribution from -10 to $+10$ where the people have rated the jokes. It is observed that there are above 4000 users who have rated $+10$ in the dataset out of 1048575 ratings.

The wikilens dataset is shown in Figure 2 where the users are the people and the items are the wikipedia articles. The rating distribution is given from 0.5 to 5, where the users rated the items in wikipedia. From the plot, it is observed that a higher rating of four is given by 5721 users, and a low rating of 1.5 is given by 730 users out of 26936 ratings. In the dataset, good books people are the users, and the items are the books having a rating distribution from 1 to 5. The rating is given by the users on different books. From the plot, we can observe that 376467 of the users have given a higher rating of four for the books, and a low rating of one is given by 24676 users out of 1048575 ratings. For the cell phone recommendation dataset, the users are the people and the items are the cell phone id's having a rating distribution from 1 to 10. The ratings are given by people on different cell phone id's. From the plot, it is observed that a higher rating of eight is given by 196 users, and a low rating of three is given by 30 users out of 990 ratings.

In community detection algorithms, modularity plays a significant impact in determining the standard of the communities that are formed. Girvan and Neuman introduced the modularity measure to assess the standards of the communities that are formed and defined modularity as

$$G = \frac{1}{2m} \sum_{(i,j)} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta\left(c_i, c_j\right) \tag{10}$$

where m is the total number of edges in the graph, A_{ij} is the edge weight between the nodes i and j, k_i and k_j are the degree of nodes i and j respectively, c_i, c_j are the communities to which nodes i and j belongs and, $\delta(c_i, c_j)$ is the Kronecker delta function which is equal to 1 if $c_i = c_j$ and 0 otherwise. The term $[A_{ij} - \frac{k_i k_j}{2m}]$ represents the difference between the observed number of edges between the nodes i and j and the expected number of edges under a null model where edges are placed at random. The values of modularity(G) varies between -1 to $+1$.

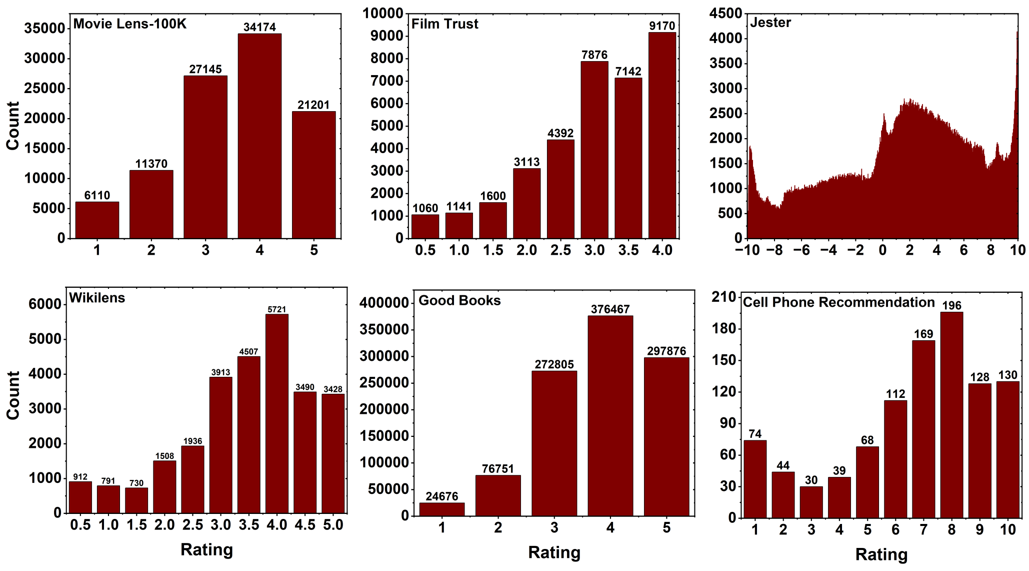

Figure 2. Rating plots distribution for six different datasets namely Movie Lens-100K, Film Trust, Jester, Wikilens, Good Books, and Cell Phone Recommendation

Figure 3 shows the modularity score that is obtained from six different datasets namely Movie Lens-100K, Film Trust, Jester, Wikilens, Good Books, and Cell Phone Recommendation for Louvain algorithm. It is observed from all the datasets that, the modularity score is increasing as the communities are increased and maintained constant after a certain number of communities. The highest modularity score is maintained by the wikilens dataset, and the lowest modularity score is given by the movie lens-100K dataset. The time taken for calculating the modularity score is shown in Table 2.

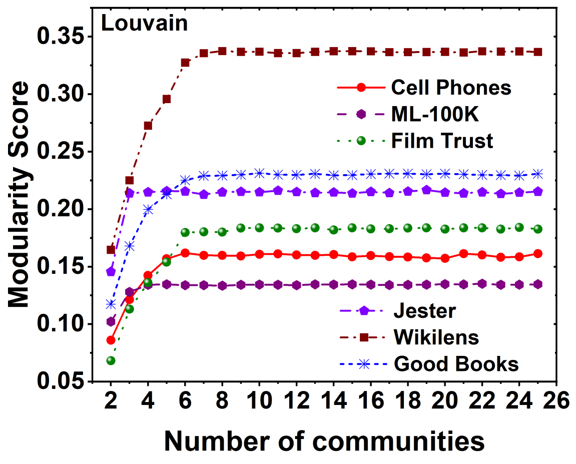

Figure 3. Modularity performance for six different datasets namely Movie Lens-100K, Film Trust, Jester, Wikilens, Good Books, and Cell Phone Recommendation for the Louvain algorithm.

Table 2. Time taken (in seconds) to calculate dividing community modularity score for six different datasets namely Movie Lens-100K, Film Trust, Jester, Wikilens, Good Books, and Cell Phone Recommendation datasets.

Dataset	Time (in sec)
Movie Lens-100K	1473.21
Film Trust	936.00
Jester	11714.99
Wikilens	1068.41
Good Books	43125.11
Cell Phone Recommendation	15.58

7. Result Analysis

This section presents a series of experiments designed to demonstrate the validity of the hypothesis proposed in the paper. Using the Louvain community detection algorithm, three different matrix factorization methods (MF, SVD++, and FANMF) are employed to predict the missing entries in the rating matrix across six distinct datasets as shown in Table 1.

The computation of the root mean square error (RMSE) involves varying the number of latent features at values of $k = 2, 10, 20, 30, 50$. Considering the random nature of the Louvain algorithm, each MF method is iterated for 25 communities and 25 iterations. The evaluation of different patterns in RMSE values, along with the total time taken to assess the MF method for six datasets is presented. The resulting graph depicts the relationship between the number of communities on the X-axis and the corresponding RMSE value on the Y-axis. Given that the cell phone recommendation dataset comprises only 99 users and 33 items, the value of k should be the minimum of number of users and items. Therefore, the iteration for k is limited to 30.

Figure 4 illustrates the variation in RMSE values across six distinct datasets when employing the MF method with different k values on a set of 25 communities. A notable observation from the graph is that, in all datasets, the absence of the Louvain algorithm results in the highest RMSE value when $c = 1$. However, as the number of communities increases through the utilization of the Louvain algorithm i.e., $c = 2, 3, \cdots, 25$, a decline in the RMSE value is observed. Notably, the movie lens-100K, good books, and cell phone recommendation datasets exhibit an increase in the RMSE value with higher k values. Conversely, in the film trust and jester datasets, the RMSE value stabilizes after reaching a certain number of communities, regardless of the k value. Furthermore, in the wikilens dataset, there exhibit a constant relationship between the RMSE value for different communities as well as the k values. Overall, it is evident that the MF method consistently yields better RMSE values with lower k values across all datasets.

Figure 5 displays the computational time required to execute the MF method on six distinct datasets, employing different k values across a set of 25 communities. Across all datasets, a noticeable observation is that the computation time for the MF method, without the utilization of community division, exceeds that of the method with community division. As the number of communities increases, a reduction in computational time is observed. Interestingly, the evaluation time of the method remains constant as the number of communities expands, regardless of the specific k values employed.

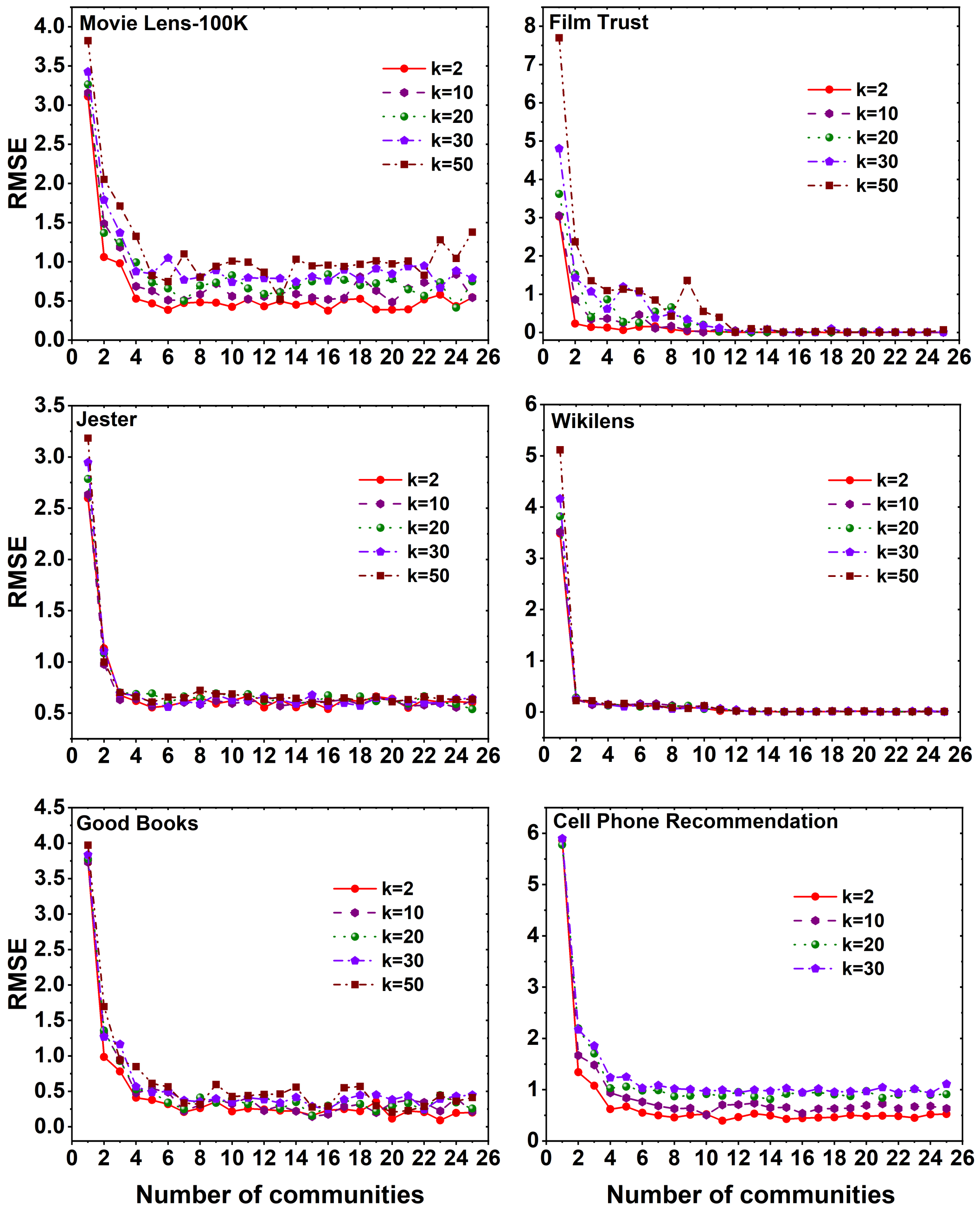

Figure 4. RMSE plots using MF method for six different datasets namely Movie Lens-100K, Film Trust, Jester, Wikilens, Good Books, and Cell Phone Recommendation with 25 communities and different k values.

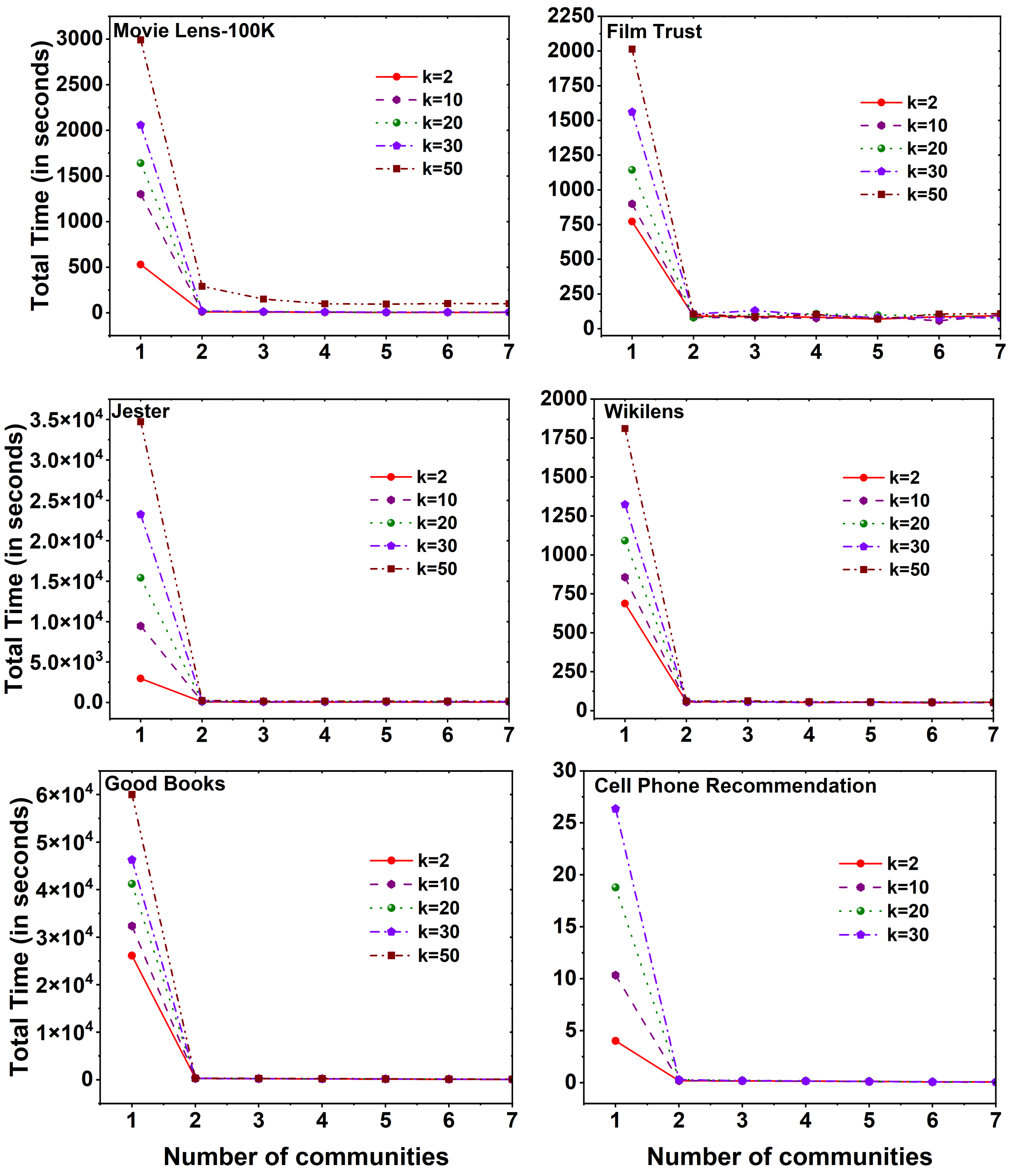

Figure 5. Time taken to calculate MF method for six different datasets namely Movie Lens-100K, Film Trust, Jester, Wikilens, Good Books, and Cell Phone Recommendation with 25 communities.

Figure 6 presents the RMSE values across six distinct datasets when employing the SVD++ method with varying k values on a set of 25 communities. Notably, at $c = 1$, a significantly high RMSE value is observed across all datasets. However, as the communities are divided, a decline in the RMSE value is observed, reaching a stable state after a certain number of communities. It is worth noting that in the SVD++ method, subtle variations in the RMSE value can be observed for different k values across all datasets.

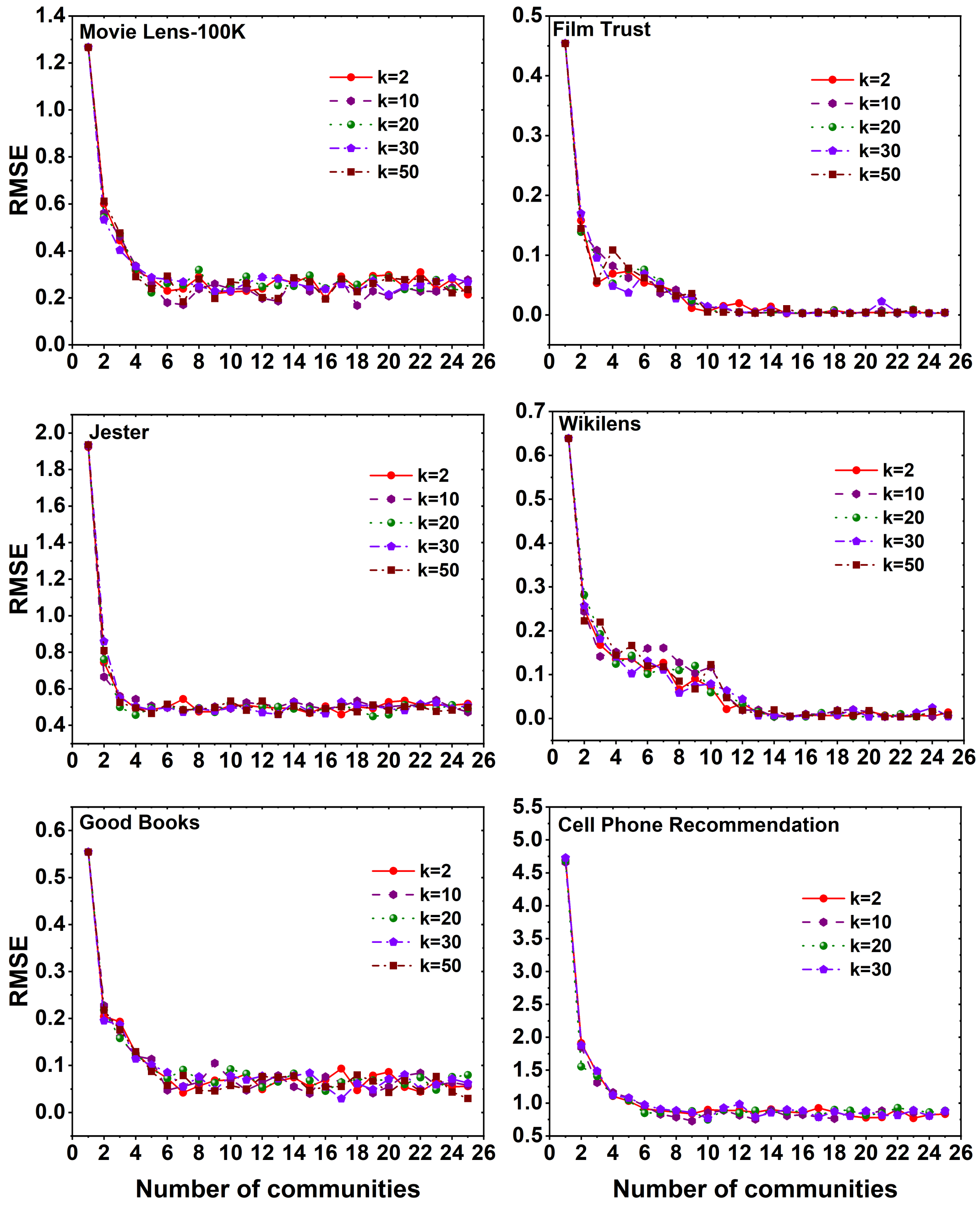

Figure 6. RMSE plots using SVD++ method for six different datasets namely Movie Lens-100K, Film Trust, Jester, Wikilens, Good Books, and Cell Phone Recommendation with 25 communities.

Figure 7 displays the computational time required to execute the SVD++ method on six distinct datasets, employing different k values across a set of 25 communities. Noteworthy observations can be made across various datasets. In the movie lens-100K and good books datasets, an increase in the number of communities leads to a decrease in the evaluation time of the method. The maximum evaluation time is observed at $c = 1$. Conversely, in the jester dataset, the maximum time is taken to evaluate the SVD++ method at c value 1, with slight fluctuations in time as the communities increase. Furthermore, for the film trust, wikilens, and cell phone recommendation datasets, it is observed that initially, evaluating the method without dividing into communities requires less time compared to the division into communities. However, after reaching a certain number of community divisions (i.e., $k = 18$ for film trust and wikilens datasets), the evaluation time becomes less than that at the time taken without community division and remains constant. Similarly, in the case of the cell phone recommendation dataset, the evaluation time becomes less than that at c value 1, and starting from c value 7 constant time is maintained for evaluation.

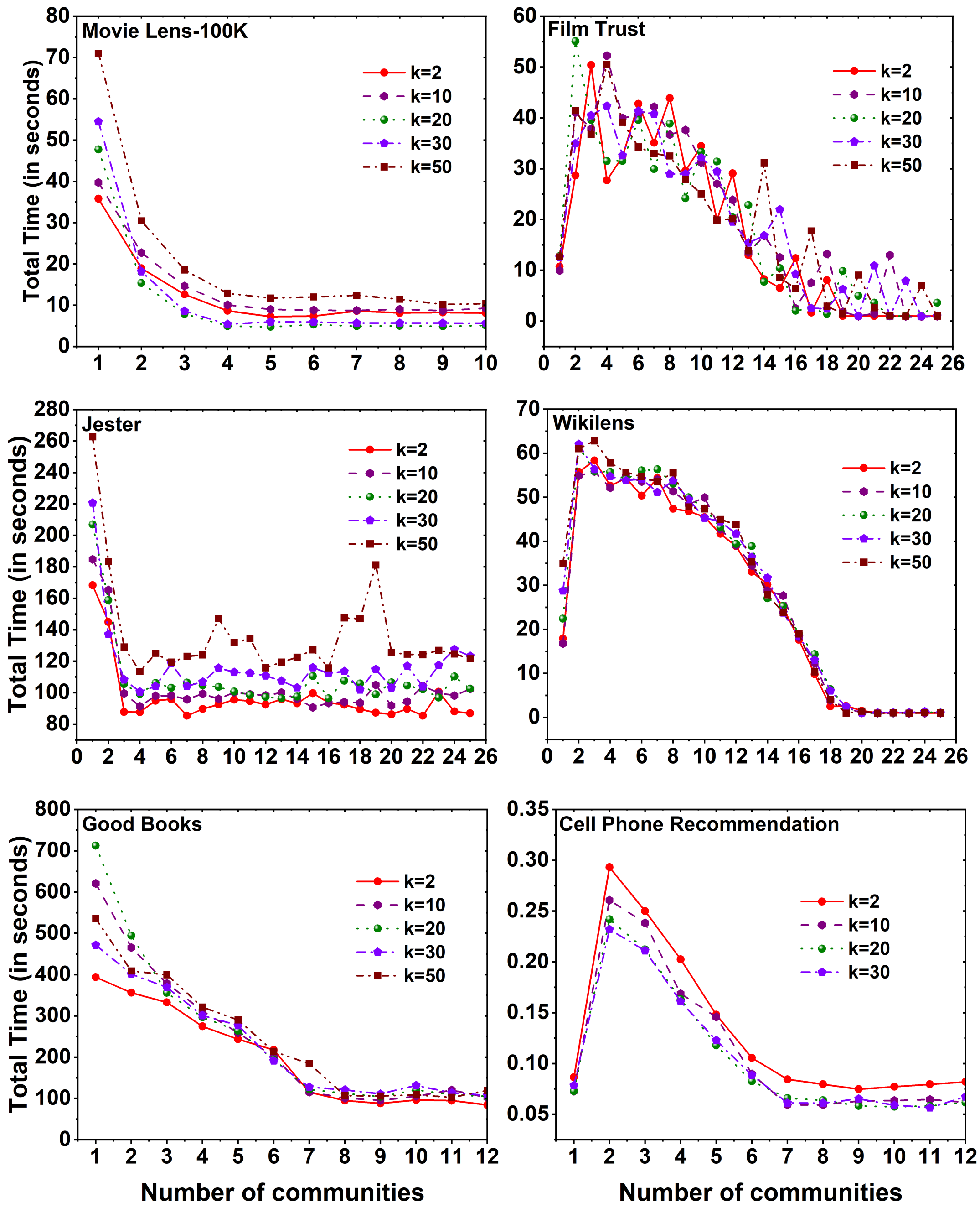

Figure 7. Time taken to calculate SVD++ method for six different datasets namely Movie Lens-100K, Film Trust, Jester, Wikilens, Good Books, and Cell Phone Recommendation with 25 communities.

The obtained RMSE values from applying the FANMF method to six distinct datasets, using different values of k across a set of 25 communities are illustrated in Figure 8. In all datasets, it is evident that increasing the k value results in a decrease in the RMSE, regardless of the specific communities. Significantly, in the film trust, wikilens, and cell phone recommendation datasets, it is notable that the RMSE values stabilize after a certain number of communities. This stability is observed across all k values, indicating a consistent RMSE value. Moreover, across all datasets, it can be observed that a higher number of latent features leads to lower RMSE values when employing the FANMF method.

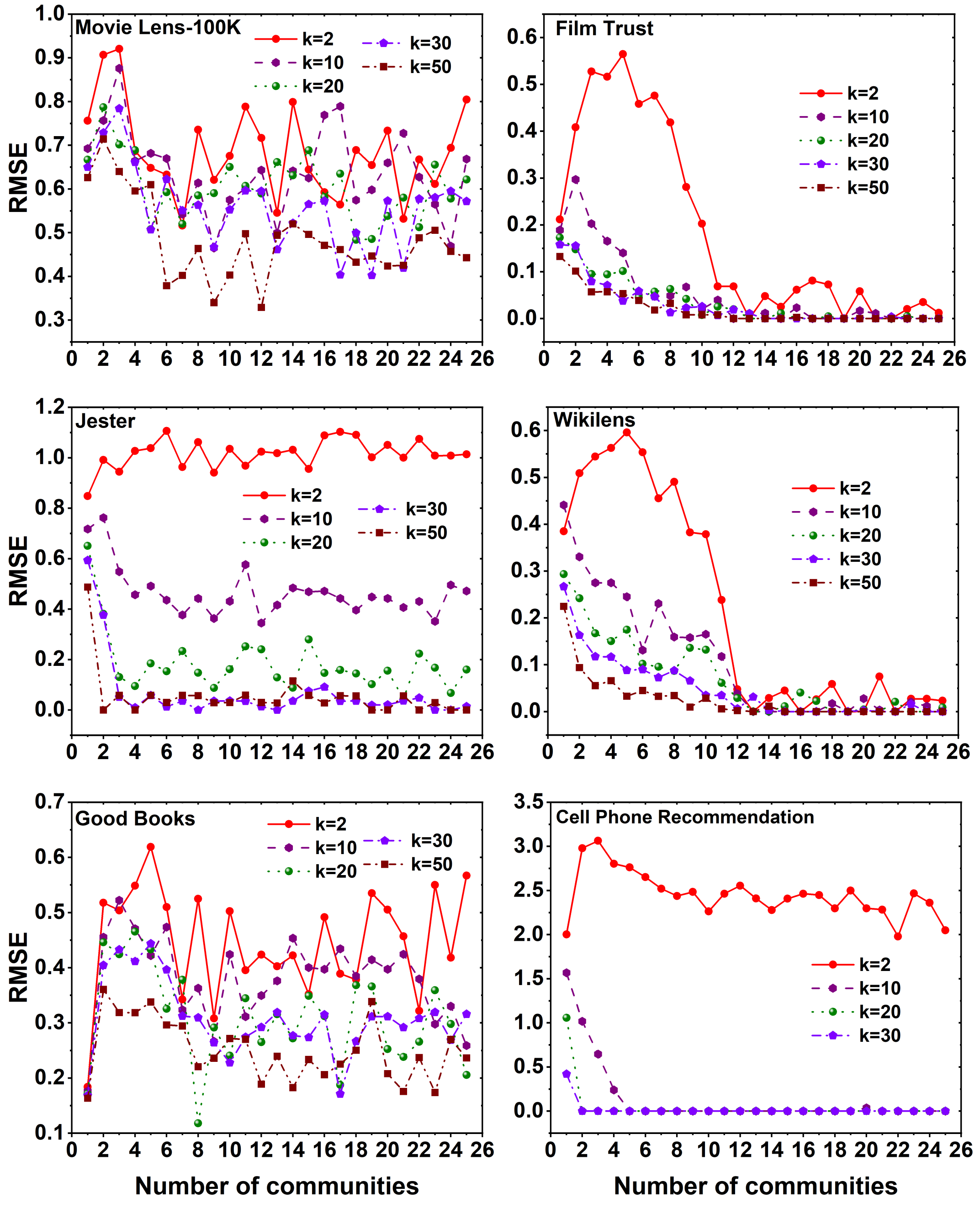

Figure 8. RMSE plots using FANMF method for six different datasets namely Movie Lens-100K, Film Trust, Jester, Wikilens, Good Books, and Cell Phone Recommendation with 25 communities.

The computational time needed to run the FANMF method on six different datasets using various k values over a set of 25 communities is shown in Figure 9. A notable observation across all datasets is that the computation time for the community 2 exceeds that of the community 1. However, after reaching a certain number of communities, the computation time decreases compared to the $c = 1$ community for the film trust, wikilens, and good books datasets. Conversely, for the movie lens-100K and jester datasets, a constant computation time is maintained, which is higher than that of the $c = 1$ community.

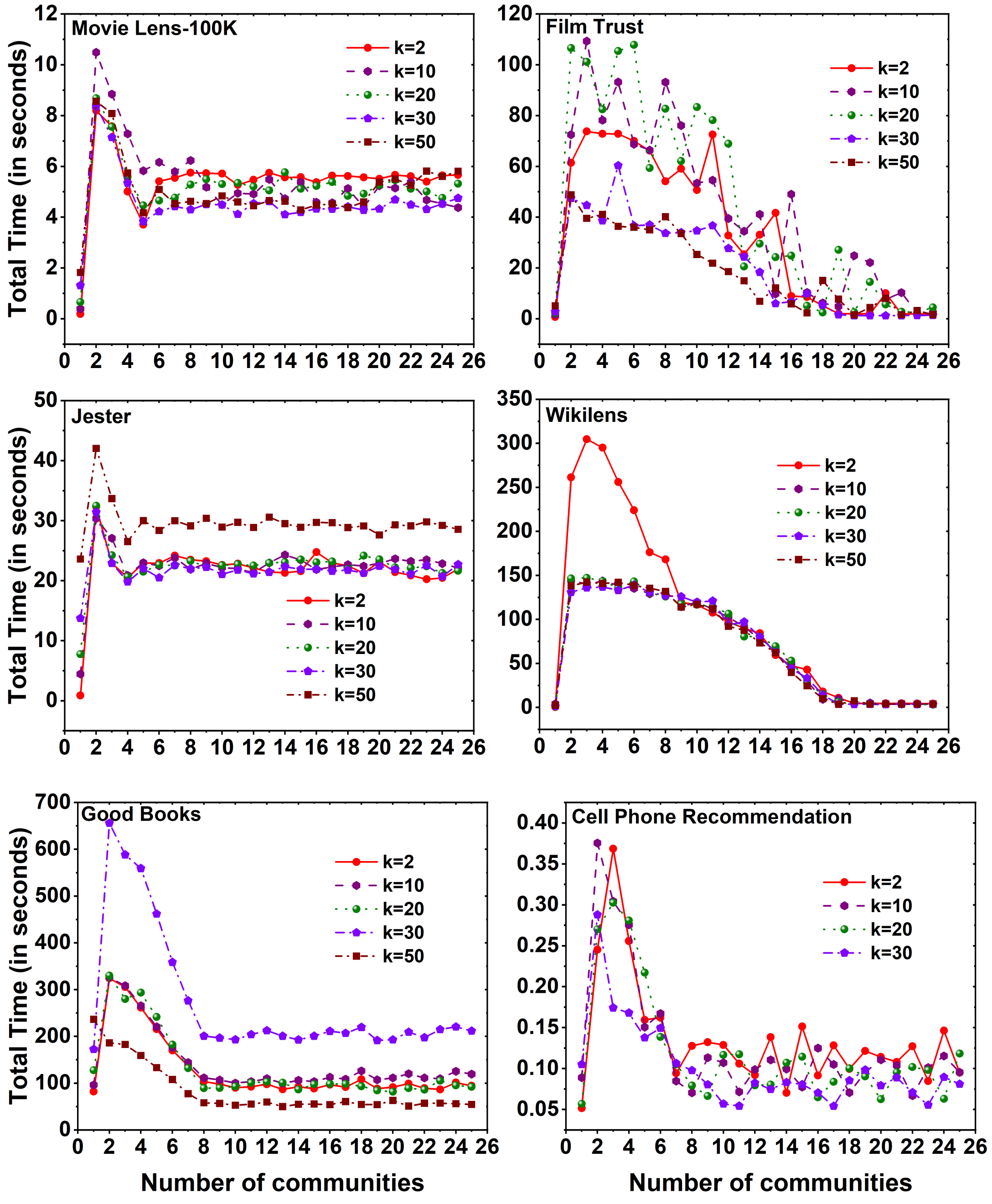

Figure 9. Time taken to calculate FANMF method for six different datasets namely Movie Lens-100K, Film Trust, Jester, Wikilens, Good Books, and Cell Phone Recommendation with 25 communities.

8. Conclusions

In this study, we have delved into the realm of parallel computation. We proposed Community based Matrix Factorization(CBMF), which parallelizes matrix factorization utilizing community information. The experimental results obtained from our study provide compelling evidence of the scalability and speedup attained through CBMF, surpassing the performance of sequential implementations. These findings underscore the immense potential of parallel computation in effectively tackling the resources in the distributed environment. Looking ahead, the introduction of our parallel computation framework unlocks fresh opportunities for the efficient processing of large datasets. It empowers researchers and practitioners to extract valuable insights from intricate networks. This framework serves as a stepping stone for future advancements in providing accurate recommendations across diverse domains. It lays the foundation for future innovations and break-through in the realm of data-driven solutions.

Author Contributions: Concept and methodology by ST, MKE and TJL. ST compiled the primary data, implemented the model, and ran the simulations. Findings, Validation, and analysis by ST, MKE and TJL. ST and MKE wrote the manuscript. All authors have read and approved to the published version of the manuscript.

Funding: There is no external funding for this work.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: All the authors stated there is no conflict of interest.

Abbreviations

Abbreviations used in this paper are as follows:

BG	Bipartite Graph
DMF	Deep Matrix Factorization
FANMF	Factorized Asymmetric Non-Negative Matrix Factorization
MF	Matrix Factorization
NLP	Natural Language Processing
NMF	Non-Negative Matrix Factorization
PCSNMF	Pairwisely Constrained Nonnegative Symmetric Matrix Factorization
RMSE	Root Mean Square Error
SVD++	Advanced Singular Value Decomposition

References

1. Adomavicius, G.; Tuzhilin, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering* **2005**, *17*, 734–749.
2. Felfernig, A.; Jeran, M.; Ninaus, G.; Reinfrank, F.; Reiterer, S.; Stettinger, M. Basic approaches in recommendation systems. *Recommendation Systems in Software Engineering* **2014**, pp. 15–37.
3. Hintz, J. Matrix Factorization for Collaborative Filtering Recommender Systems **2015**.
4. kumar Bokde, D.; Girase, S.; Mukhopadhyay, D. Role of matrix factorization model in collaborative filtering algorithm: A survey. *CoRR, abs/1503.07475* **2015**.
5. Koren, Y.; Bell, R.; Volinsky, C. Matrix factorization techniques for recommender systems. *Computer* **2009**, *42*, 30–37.
6. Mehta, R.; Rana, K. A review on matrix factorization techniques in recommender systems. 2017 2nd International Conference on Communication Systems, Computing and IT Applications (CSCITA). IEEE, 2017, pp. 269–274.
7. Abdrabbah, S.B.; Ayachi, R.; Amor, N.B. Collaborative filtering based on dynamic community detection. *Dynamic Networks and Knowledge Discovery* **2014**, *85*.

8. Kumar, P.; Chawla, P.; Rana, A. A review on community detection algorithms in social networks. 2018 4th International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT). IEEE, 2018, pp. 304–309.
9. Bedi, P.; Sharma, C. Community detection in social networks. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **2016**, *6*, 115–135.
10. Du, N.; Wu, B.; Pei, X.; Wang, B.; Xu, L. Community detection in large-scale social networks. Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis, 2007, pp. 16–25.
11. Karataş, A.; Şahin, S. Application areas of community detection: A review. 2018 International congress on big data, deep learning and fighting cyber terrorism (IBIGDELFT). IEEE, 2018, pp. 65–70.
12. Lalwani, D.; Somayajulu, D.V.; Krishna, P.R. A community driven social recommendation system. 2015 IEEE International conference on big data (big data). IEEE, 2015, pp. 821–826.
13. Guo, W.; Gao, H.; Shi, J.; Long, B.; Zhang, L.; Chen, B.C.; Agarwal, D. Deep natural language processing for search and recommender systems. Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 3199–3200.
14. Musto, C.; de Gemmis, M.; Lops, P.; Semeraro, G. Generating post hoc review-based natural language justifications for recommender systems. *User Modeling and User-Adapted Interaction* **2021**, *31*, 629–673.
15. Chen, S.; Owusu, S.; Zhou, L. Social network based recommendation systems: A short survey. 2013 international conference on social computing. IEEE, 2013, pp. 882–885.
16. Sun, Z.; Han, L.; Huang, W.; Wang, X.; Zeng, X.; Wang, M.; Yan, H. Recommender systems based on social networks. *Journal of Systems and Software* **2015**, *99*, 109–119.
17. Kim, J.K.; Cho, Y.H.; Kim, W.J.; Kim, J.R.; Suh, J.H. A personalized recommendation procedure for Internet shopping support. *Electronic commerce research and applications* **2002**, *1*, 301–313.
18. Wei, K.; Huang, J.; Fu, S. A survey of e-commerce recommender systems. 2007 international conference on service systems and service management. IEEE, 2007, pp. 1–5.
19. Kim, K.j.; Ahn, H. A recommender system using GA K-means clustering in an online shopping market. *Expert systems with applications* **2008**, *34*, 1200–1209.
20. Hasan, M.R.; Jha, A.K.; Liu, Y. Excessive use of online video streaming services: Impact of recommender system use, psychological factors, and motives. *Computers in Human Behavior* **2018**, *80*, 220–228.
21. Park, D.H.; Kim, H.K.; Choi, I.Y.; Kim, J.K. A literature review and classification of recommender systems research. *Expert systems with applications* **2012**, *39*, 10059–10072.
22. Isinkaye, F.O. Matrix factorization in recommender systems: algorithms, applications, and peculiar challenges. *IETE Journal of Research* **2021**, pp. 1–14.
23. Hallinan, B.; Striphas, T. Recommended for you: The Netflix Prize and the production of algorithmic culture. *New media & society* **2016**, *18*, 117–137.
24. Paatero, P.; Tapper, U. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics* **1994**, *5*, 111–126.
25. Mnih, A.; Salakhutdinov, R.R. Probabilistic matrix factorization. *Advances in neural information processing systems* **2007**, *20*.
26. Mastorakis, N.E. The singular value decomposition (svd) in tensors (multidimensional arrays) as an optimization problem. solution via genetic algorithms and method of nelder-mead. *WSEAS Transactions on Systems* **2007**, *6*, 17–23.
27. Hu, Y.; Koren, Y.; Volinsky, C. Collaborative filtering for implicit feedback datasets. 2008 Eighth IEEE international conference on data mining. Ieee, 2008, pp. 263–272.
28. Shi, X.; Lu, H.; He, Y.; He, S. Community detection in social network with pairwisely constrained symmetric non-negative matrix factorization. Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015, 2015, pp. 541–546.
29. Xue, H.J.; Dai, X.; Zhang, J.; Huang, S.; Chen, J. Deep matrix factorization models for recommender systems. IJCAI. Melbourne, Australia, 2017, Vol. 17, pp. 3203–3209.
30. Schlichtkrull, M.; Kipf, T.N.; Bloem, P.; Van Den Berg, R.; Titov, I.; Welling, M. Modeling relational data with graph convolutional networks. The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15. Springer, 2018, pp. 593–607.

31. Tosyali, A.; Kim, J.; Choi, J.; Jeong, M.K. Regularized asymmetric nonnegative matrix factorization for clustering in directed networks. *Pattern Recognition Letters* **2019**, *125*, 750–757.
32. Girvan, M.; Newman, M.E. Community structure in social and biological networks. *Proceedings of the national academy of sciences* **2002**, *99*, 7821–7826.
33. Raghavan, U.N.; Albert, R.; Kumara, S. Near linear time algorithm to detect community structures in large-scale networks. *Physical review E* **2007**, *76*, 036106.
34. Blondel, V.D.; Guillaume, J.L.; Lambiotte, R.; Lefebvre, E. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* **2008**, *2008*, P10008.
35. Pons, P.; Latapy, M. Post-processing hierarchical community structures: Quality improvements and multi-scale view. *Theoretical Computer Science* **2011**, *412*, 892–900.
36. Singh, A.P.; Gordon, G.J. A unified view of matrix factorization models. Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2008, Antwerp, Belgium, September 15-19, 2008, Proceedings, Part II 19. Springer Berlin Heidelberg, 2008, pp. 358–373.
37. Ivarsson, J.; Lindgren, M. Movie recommendations using matrix factorization, 2016.
38. Kumar, R.; Verma, B.; Rastogi, S.S. Social popularity based SVD++ recommender system. *International Journal of Computer Applications* **2014**, *87*.
39. Rendle, S. Factorization machines. 2010 IEEE International conference on data mining. IEEE, 2010, pp. 995–1000.
40. Hosseinzadeh Aghdam, M.; Daryaie Zanjani, M. A novel regularized asymmetric non-negative matrix factorization for text clustering **2021**.
41. Lee, D.D.; Seung, H.S. Learning the parts of objects by non-negative matrix factorization. *Nature* **1999**, *401*, 788–791.
42. Alzahrani, T.; Horadam, K.J. Community detection in bipartite networks: Algorithms and case studies. In *Complex systems and networks: Dynamics, controls and applications*; Springer, 2015; pp. 25–50.
43. Gupta, S.; Singh, D.P. Recent trends on community detection algorithms: A survey. *Modern Physics Letters B* **2020**, *34*, 2050408.
44. He, X.; Zhang, H.; Kan, M.Y.; Chua, T.S. Fast matrix factorization for online recommendation with implicit feedback. Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, 2016, pp. 549–558.
45. Seifikar, M.; Farzi, S.; Barati, M. C-blondel: an efficient louvain-based dynamic community detection algorithm. *IEEE Transactions on Computational Social Systems* **2020**, *7*, 308–318.