




# Experimental Analysis of Pipelining Community Detection and Recommender Systems

Ryan Dutra de Abreu<sup>1</sup><sup>a</sup>, Laura Silva de Assis<sup>1</sup><sup>b</sup> and Douglas O. Cardoso<sup>2</sup><sup>c</sup>

<sup>1</sup>*Celso Suckow da Fonseca, Federal Center of Technological Education, Rio de Janeiro, RJ, Brazil*

<sup>2</sup>*Smart Cities Research Center, Polytechnic Institute of Tomar, Tomar, Portugal*

**Keywords:** Collaborative Filtering, Social Networks, User Profiling, Collective Behavior.

**Abstract:** Community detection and recommender systems are two subjects of the highest relevance among data-oriented computational methods, considering their current applications in various contexts. This work investigated how pipelining these tasks may lead to better recommendations than those obtained without awareness of implicit communities. We experimentally assessed various combinations of methods for community detection and recommendation algorithms, as well as synthetic and real datasets. This targeted to unveil interesting patterns in the behavior of the resulting systems. Our results show that insights into communities can significantly improve both the effectiveness and efficiency of recommendation algorithms in some favorable scenarios. These findings can be used to help data science researchers and practitioners to better understand the benefits and limitations of this methodology.

## 1 INTRODUCTION

The Internet's significant growth in the last three decades has transformed how society conducts business and communicates (Castells and Cardoso, 1996). By 2020, its widespread usage had become a global norm (Roser et al., 2015), resulting in new relationship structures, habits, consumption patterns, and markets (Paul and Aithal, 2018).

Although beneficial in many aspects, this digital evolution has led to information overload for users, making it challenging to find relevant data and make informed decisions (Dias, 2014). In this context, recommendation systems offer promise in managing this overload, simplifying decision-making by suggesting relevant items (Panteli et al., 2019).


Grounded in cognitive science and information retrieval, these systems are crucial for generating personalized suggestions based on user preferences (Rich, 1979; Sanderson and Croft, 2012). They can employ various approaches, including collaborative filtering, content-based filtering, and hybrid filtering (Burke, 2000; Adomavicius and Tuzhilin, 2005; Pazzani and Billsus, 2007; Schafer et al., 2007).


Collaborative filtering (CF) techniques are widely used due to its efficiency, recommending items based on similar users' preferences (Gorripati and Vatsavayi, 2017). Yet, many commercial systems rely on large, sparse user-item matrices, posing challenges such as the "cold-start" problem (Guo, 2013).


To tackle these challenges, researchers have explored strategies like incorporating additional data to model user preferences, introducing new similarity measures, and using community detection algorithms to uncover network structures (Guo, 2013; Keerthi Gorripati and Angadi, 2018; Xie and Szymanski, 2013; Bourhim et al., 2018; Bhaskaran et al., 2022). While integrating community detection with recommender systems has shown promise in enhancing recommendation accuracy, there is still a wide range of experimental opportunities in the field.

This study investigates combining these tasks to improve recommendations through implicit community awareness. We explore novel algorithm combinations, evaluating their efficiency and effectiveness with synthetic and real datasets to provide insights for data science researchers and practitioners.

Remaining sections include an overview of related literature (Section 2), details of the community-oriented recommendation methodology (Section 3), experimental descriptions and results (Section 4), and a research summary (Section 5).

<sup>a</sup> <https://orcid.org/0009-0007-8339-1631>

<sup>b</sup> <https://orcid.org/0000-0003-3081-9722>

<sup>c</sup> <https://orcid.org/0000-0002-1932-334X>

## 2 RELATED WORK

Integrating community detection algorithms with recommender systems has gained recent attention. These approaches use data structure and relationships to enhance recommendation accuracy and diversity. This section provides an overview of relevant works in this area.

Pham et al. (Pham et al., 2011) proposed a clustering approach that improves recommendation quality by incorporating social information. Their user-based algorithms make predictions by aggregating ratings from the active user’s nearest neighbors with weights. The technique outperformed traditional CF methods in evaluations using DBLP and Epinion data.

A graph-based CF approach treating users as a network of highly similar sub-communities was presented in (Bourhim et al., 2018; Bourhim et al., 2019). By utilizing key-nodes and the Louvain modularity-based algorithm to group users, significant performance gains were achieved compared to classical recommendation approaches.

Çiftçi et al. (Çiftçi et al., 2021) used Louvain and Label Propagation community detection algorithms to identify artist clusters, aligning well with Spotify’s artist lists. Additionally, Kherad and Bidgoly introduced a recommendation method (Kherad and Bidgoly, 2022) combining matrix factorization, graph analysis, and deep learning techniques. They employed the Louvain algorithm for community detection and HITS for identifying important nodes in the user communication network, resulting in superior performance compared to existing methods.

In (Patra et al., 2014), a similarity-based CF method was introduced to address the user cold-start issue in sparse data situations, delivering reliable recommendations for new users with limited ratings, substantially improving accuracy compared to existing solutions.

Similarly, Gorripati and Vatsavayi (Gorripati and Vatsavayi, 2017) proposed a community-based CF approach that tackles both data sparsity and cold-start problems. By examining user preferences within the same community, they enhance recommendation quality. Their method involves identifying user patterns, creating a network based on co-rating relationships, identifying community structures, and computing neighborhood similarity.

In the realm of music recommendation systems, (Cao et al., 2020) addressed data sparsity and cold-start issues. The authors introduced a community detection algorithm based on the Louvain method, which exhibited higher accuracy, stability, and shorter running times compared to alternative ap-

proaches, validating the effectiveness of their cold-start algorithm.

While these works have shown the potential of integrating community detection algorithms with recommender systems to enhance recommendation accuracy and address challenges, they have some limitations. Furthermore, there is still a research gap in exploring the combinations of community discovery and recommendation algorithms, as well as their impact on system performance. Our work aims to advance this integration by exploring novel algorithm combinations, assessing their effectiveness and efficiency using synthetic and real datasets, and gaining insights into system behavior. This contributes to more effective and personalized recommendation systems.

## 3 PROPOSED METHODOLOGY

This work’s core inspiration is to enhance recommendation algorithms using community detection techniques. This approach is highly adaptable and can be employed with a range of prediction algorithms, including heuristics, neighborhood methods, matrix factorization, and other CF techniques. The methodology involves four steps, which will be detailed in the following subsections.

### 3.1 Graph Projection

In this subsection, we outline the process of transforming a bipartite graph of user-item ratings into a unipartite weighted graph focused solely on users. This projection enables us to capture user relationships and interactions while considering the weighted connections derived from their ratings.

Consider a weighted bipartite graph  $\mathcal{G} = (\mathcal{U}, \mathcal{I}, \mathcal{E}, r)$  where  $\mathcal{U}$  is the set of users nodes,  $\mathcal{I}$  is the set of items nodes representing the catalog items,  $\mathcal{E}$  is the set of edges connecting nodes from  $\mathcal{U}$  to  $\mathcal{I}$  and  $r$  represents the ratings, i.e., the weights assigned to each edge. It can be evaluated the dissimilarity between two users  $u_k$  and  $u_l$ , where  $(u_k, u_l) \in \mathcal{U}^2$ , by iterating over the items that both users have rated and evaluating the mean squared difference between their corresponding ratings. This will result a unipartite weighted graph  $\mathcal{G}' = (\mathcal{U}', \mathcal{E}', w)$ , where  $\mathcal{U}'$  represents the set of nodes in the weighted projection – which corresponds to the set of nodes  $\mathcal{U}$  in the bipartite graph,  $\mathcal{E}'$  represents the set of edges in the weighted projection, capturing the connections between pairs of nodes in  $\mathcal{U}'$  which rated at least one item in common, and  $w$  represents the weight function in the weighted projection, assigning numerical

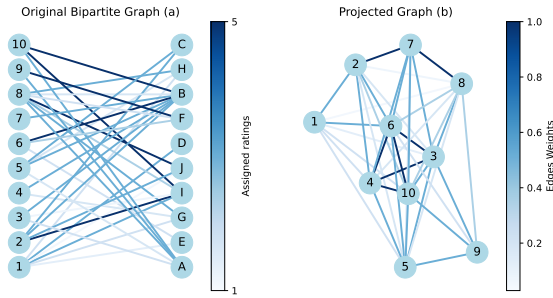


Figure 1: Example of a bipartite graph: (a) representing user-item relationships and its corresponding projection (b) showing the connections among users based on their ratings.

values to the edges in  $\mathcal{E}'$ .

Mathematically, the weight  $w_{u_k, u_l}$  is defined as shown in Equation (1):

$$w_{u_k, u_l} = \frac{1}{|\mathcal{G}[u_k] \cap \mathcal{G}[u_l]|} \left( \sum_{i \in \mathcal{G}[u_k] \cap \mathcal{G}[u_l]} (r_{u_k, i} - r_{u_l, i})^2 \right) \quad (1)$$

where,  $\mathcal{G}[u_k]$  and  $\mathcal{G}[u_l]$  represent the sets of items rated by users  $u_k$  and  $u_l$ , respectively.  $r_{u_k, i}$  and  $r_{u_l, i}$  represent the ratings of  $i$ -th item given by users  $u_k$  and  $u_l$ , respectively. To give the just mentioned weights a positive connotation, we apply the transformation shown in Equation (2), which also acts as a normalization to the  $(0, 1]$  interval:

$$w'_{u_k, u_l} = \frac{1}{1 + w_{u_k, u_l}} \quad (2)$$

Larger  $w$  values result in smaller normalized weights  $w'$ , indicating lower user similarity, while smaller  $w$  values lead to larger normalized weights, indicating higher similarity. These weighted connections reflect user relationships in the bipartite graph and serve as the basis for constructing the projected user-user graph. This transformation helps us focus on user-user relationships for recommendation purposes.

Figure 1 visually illustrates a bipartite graph  $\mathcal{G}$  and its corresponding projection  $\mathcal{G}'$ , generated from a fictitious dataset of 10 users and 10 items with ratings from 1 to 5. The left side shows user-item relationships, while the right side depicts connections among users based on their ratings.

### 3.2 Community Detection

Community detection aims to identify clusters of users within the graph who exhibit similar patterns or behavior. By uncovering these communities, we can

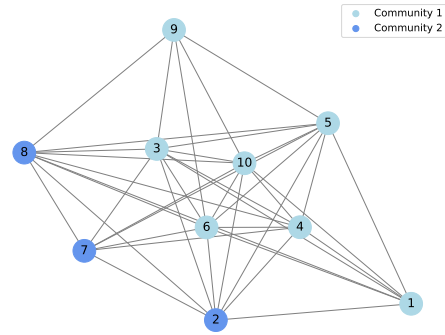


Figure 2: Graph projection with identified communities.

gain insights into the underlying structure and dynamics of user interactions, which can further enhance the effectiveness of our recommendation algorithm.

To detect communities in the projected graph, we can employ various community discovery algorithms in order to identify densely connected groups (Choudhary et al., 2023). In the context of recommendation systems, particular algorithms have been widely used for community detection, like Louvain (Blondel et al., 2008), Girvan-Newman (Girvan and Newman, 2002), Infomap (Rosvall and Bergstrom, 2008) and Label Propagation (Raghavan et al., 2007).

This clustering process is illustrated in Figure 2, where the Louvain algorithm is applied to the projected graph generated in the previous subsection (Figure 1(b)). This application results in a graph visualization where each node is assigned a specific color representing its community membership.

One can gain valuable insights into the users' collective behavior through community detection techniques, leading to a more personalized and satisfactory user experience. Once communities are detected, they represent user subsets with shared characteristics, allowing for personalized recommendations based on community affiliations.

### 3.3 Model Instantiation and Training

This section focuses on enhancing the recommendation model training phase using community information. After completing graph projection and community detection, we aim to incorporate community insights to capture collective user behavior and preferences.

Let us denote the training dataset as  $\mathcal{D}$ , the recommendation algorithm as  $\mathcal{A}$ , and the set of communities as  $\mathcal{C}$ . The community-based recommendation model training process can be outlined as follows. Consider a dataset subset  $\mathcal{D}_i$  is created for each community  $C_i \in \mathcal{C}$ , containing ratings and preferences of users in that community. Using  $\mathcal{D}_i$ , the recommen-

dation model  $M_i$  which is associated with community  $C_i$  can be trained using the recommendation algorithm  $\mathcal{A}$ . The model  $M_i$  learns from the user-item interactions within  $\mathcal{D}_i$ , capturing the preferences and patterns specific to community  $C_i$ .

After training the model  $M_i$ , it is stored as the trained model specific to community  $C_i$ . All this process is repeated for each community  $C_i \in \mathcal{C}$ , resulting in trained recommendation models  $\mathcal{M} = \{M_1, M_2, \dots, M_{|\mathcal{C}|}\}$  for each of all  $|\mathcal{C}|$  communities. The Algorithm 1 summarizes the steps involved in community-based recommendation model training.

```

1 for community  $C_i \in \mathcal{C}$  do
2   Prepare the training dataset  $\mathcal{D}_i$ , the subset
   of  $\mathcal{D}$  regarding users in  $C_i$ ;
3   Train the recommendation model  $M_i$ 
   using  $\mathcal{D}_i$  and algorithm  $\mathcal{A}$ ;
4   Store the trained model  $M_i$  specific to  $C_i$ ;
5 end
6 return Trained recommendation models
    $M_1, M_2, \dots, M_{|\mathcal{C}|}$  for each community

```

Algorithm 1: Community-Based-RMT( $\mathcal{D}$ ,  $\mathcal{A}$ ,  $\mathcal{C}$ ).

As an illustrative example, for the two-communities fictitious projected graph mentioned earlier (Figure 2), this iterative training process would generate two training instances: one for the first community (users 1, 3, 4, 5, 6, 9, and 10) and another for the second community (users 2, 7, and 8). By incorporating community information into training, we enhance the accuracy and effectiveness of the recommendation model. It learns not only from individual user-item interactions but also from collective behavior within each community, capturing nuanced preferences and improving overall recommendations.

### 3.4 Rating Prediction

This section covers the prediction phase of the proposed recommendation methodology. Once the list of communities and a trained model for each community is obtained, the next phase can predict recommendations for any user based on their community membership.

For a user  $u$  for whom we want to provide recommendations, we can find the community  $C_i$  to which user  $u$  belongs. Once community  $C_i$  is identified, we retrieve the trained recommendation model  $M_i$  specific to community  $C_i$ . Using model  $M_i$ , we can generate personalized recommendations for user  $u$ . The model utilizes user-item interactions and the learned patterns within the community to provide recommen-

dations tailored to the preferences and behavior of users in that community.

The prediction process involves using the trained model  $M_i$  to calculate the predicted ratings or scores for items that user  $u$  has not yet interacted with. This can be done using various techniques, such as CF, matrix factorization, or deep learning-based approaches. The specific method depends on the recommendation algorithm chosen during the training phase. Once the predicted ratings or scores are obtained, it is possible to classify the items or select the best-ranked recommendations to present to user  $u$ . These recommendations are based on the preferences and interests of users within the same community as  $u$ , capturing the collective behavior and preferences of that community. Algorithm 2 summarizes the prediction process for a single user, taking as input a user  $u$ , the set of communities  $\mathcal{C}$ , where  $\mathcal{C} = \{C_1, C_2, \dots, C_{|\mathcal{C}|}\}$ , and the trained recommendation models  $\mathcal{M} = \{M_1, M_2, \dots, M_{|\mathcal{C}|}\}$ .

```

1  $C_i \leftarrow$  findCommunity( $u$ );
2  $M_i \leftarrow$  Trained recommendation model
   specific to  $C_i$ ;
3 for each item  $i$  not rated by user  $u$  do
4    $score_i \leftarrow$  predictScore( $M_i, u, i$ );
5 end
6 Sort items by score in descending order;
7 return top- $k$  recommendations for user  $u$ 

```

Algorithm 2: Community-Based-RP( $u$ ,  $\mathcal{C}$ ,  $\mathcal{M}$ ,  $k$ ).

By leveraging community structures and trained recommendation models, we can offer personalized recommendations influenced not just by individual user behavior but also by their community's behavior. This captures community-specific nuances, leading to more accurate, relevant recommendations.

## 4 EXPERIMENTAL ASSESSMENT

This section covers experiments evaluating the proposed community-based recommendation methodology. We conducted experiments on synthetic and real datasets to gauge its effectiveness and robustness. We also compared the results achieved with this approach to default settings of established recommendation algorithms to assess the impact of community detection. For transparency and reproducibility, the source code is publicly accessible<sup>1</sup>.

<sup>1</sup><https://github.com/ryan-dutra/Recommender-System/>

## 4.1 Datasets

The experiments were conducted using both a synthetic dataset generated specifically for this study, and a well-known real-world dataset, Movielens 100k<sup>2</sup>. The choice of these datasets was motivated by the need to evaluate the performance and robustness of our community-based recommendation methodology across different data scenarios.

The synthetic dataset was generated to simulate a graph with a community structure, enabling the establishment of ground truth regarding community memberships and user-item preferences. Creating this synthetic dataset allowed for a systematic study of how community detection affects recommendation performance. The dataset generation employed a custom function for simulating a bipartite graph, incorporating parameters like the number of users, communities, items, categories, as well as connection probabilities and strengths. These parameters enabled the creation of synthetic datasets with desired community structures and interaction patterns.

To generate the communitarian recommendation graph, specific steps were followed. First, community labels were randomly assigned to users while ensuring sorted labels for consistency, representing users' community memberships. Community size distribution was calculated by counting users in each community. Similarly, category labels were randomly assigned to items, with sorted labels for consistency, and category size distribution was computed based on the assigned labels, representing item distribution across categories.

An affinity matrix  $\mathcal{L}$  was constructed to capture the likelihood of users liking items in specific categories, with entries generated randomly based on a given probability  $\alpha$ . This matrix quantified the degree of affinity between communities and categories, forming the basis for modeling user-item interactions.

To consider varying liking strengths, the affinity matrix  $\mathcal{L}$  was transformed into an adjusted affinity matrix  $\mathcal{L}'$ . Values in  $\mathcal{L}'$  were scaled by the liking strength parameter  $\beta$ , which controlled the overall impact of affinity on recommendations. Additionally, remaining values in  $\mathcal{L}'$  were adjusted to  $\frac{(1-\beta)}{2}$ , ensuring a balanced influence of affinity across communities and categories.

By combining the adjusted affinity matrix  $\mathcal{L}'$  with matrices of zeros, a probability matrix  $\mathcal{P}$  was constructed. This matrix represented the probabilities of user-item interactions based on community and category affiliations. Leveraging user and item distributions along with the probability matrix  $\mathcal{P}$ , a stochastic

block model graph (*sbg*) was created using the NetworkX library (Hagberg et al., 2008). The stochastic block model graph depicted the recommendation graph, with block sizes corresponding to user and item distributions.

## 4.2 Experimental Protocol

To evaluate our community-based recommendation methodology, we divided the datasets into training and testing sets. Various test sizes were considered, including 25%, 10%, and 1% of total ratings. We employed random permutation cross-validation to ensure reliable and unbiased evaluations. This technique involves shuffling the dataset and then splitting it into multiple training and testing subsets, mitigating potential biases due to data ordering. The scikit-learn library's random permutation cross-validation implementation was used with five splits to generate diverse training and testing subsets, ensuring robust performance assessment.

In each training instance, we followed several steps for the recommendation process. First, we transformed the bipartite graph into a projected graph, as described in Section 3, focusing on user-user and item-item relationships. Community detection algorithms, specifically Louvain (Blondel et al., 2008) and Paris (Xie et al., 2013), were applied to identify communities based on connectivity patterns between users and items.

Once communities were identified, recommendation models were instantiated and trained separately for each community. We evaluated five recommendation algorithms available in the Surprise library (Hug, 2020), categorized as follows:

1. **Matrix Factorization Algorithms:** SVD and NMF.
2. **Neighborhood-Based Algorithms:** k-NN and Slope One.
3. **Clustering Algorithm:** Co-Clustering.

Additionally, the NormalPredictor algorithm served as a random baseline. Trained models were used to generate predictions for user-item pairs in the training data, with each model specific to its corresponding community, following the approach outlined in Section 3.

To assess performance, we employed the RMSE metric, quantifying the accuracy of recommendation models in predicting user-item ratings. We also evaluated the computational cost, including training and testing times in seconds, to assess the efficiency of our community-based recommendation approach.

<sup>2</sup><https://movielens.org/>

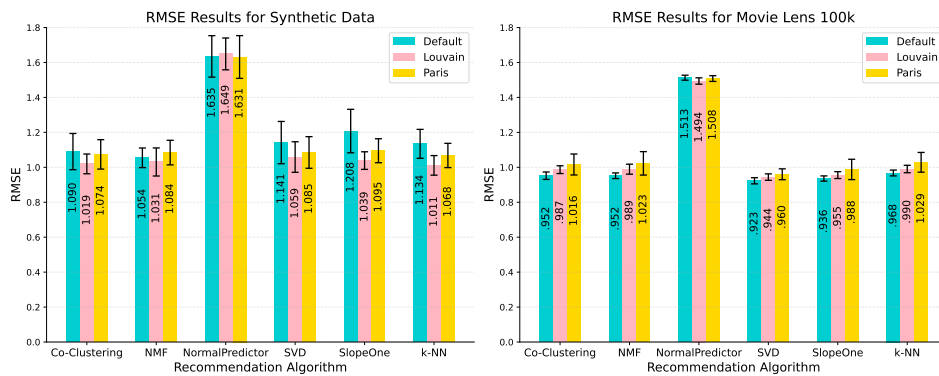


Figure 3: Predictive performance considering various datasets, community detection approaches and recommendation algorithms.

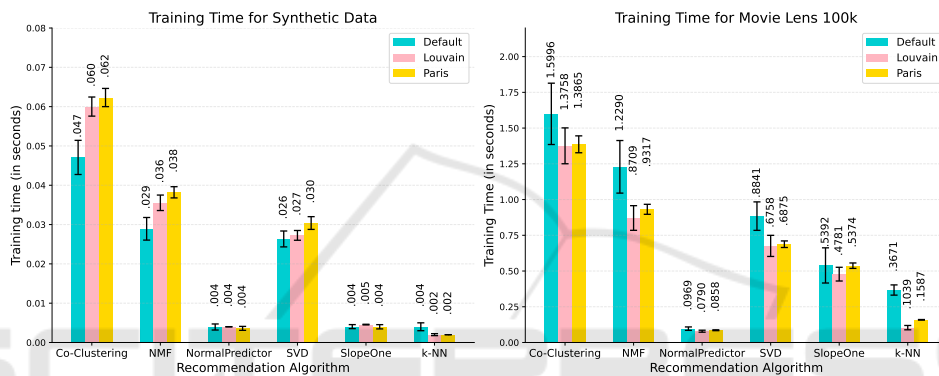


Figure 4: Training time considering various datasets, community detection approaches and recommendation algorithms.

By conducting experiments on synthetic and real-world datasets, we aimed to comprehensively evaluate our approach’s performance across various data settings. Subsequent subsections present and discuss the results obtained.

### 4.3 Results and Discussion

This section presents computational experiment results evaluating the community-based recommendation approach’s performance, comparing it with default recommendation methods, and analyzing effectiveness and computational cost across various scenarios.

To evaluate the community-based recommendation approach, tests were conducted on both synthetic and real datasets, as previously explained. Figure 3 illustrates the performance comparison between the proposed approach (utilizing Louvain or Paris algorithms for community discovery) and the default alternative (without a community discovery algorithm). The bar heights represent the average RMSE of the iterations, while the error bands show the variability or uncertainty in these averages, calculated using the

standard deviation.

In the synthetic dataset scenario, our community-based recommendation approach consistently outperformed the default method (without a community detector), especially with neighborhood-based algorithms, which showed significant performance improvement when combined with community detection.

In the real dataset scenario (Movie Lens 100k), our approach generally performed similarly to the default method with slight differences. The only exception was the Normal Predictor algorithm, which showed lower RMSE when community detectors were used.

These results indicate that community detectors can benefit recommendation tasks, especially for datasets with clear network structures. Additionally, neighborhood-based algorithms showed the most significant improvement when integrated with our community-based approach. This highlights the potential effectiveness of combining neighborhood-based algorithms with community-based recommendation strategies.

In addition to performance evaluation, the com-

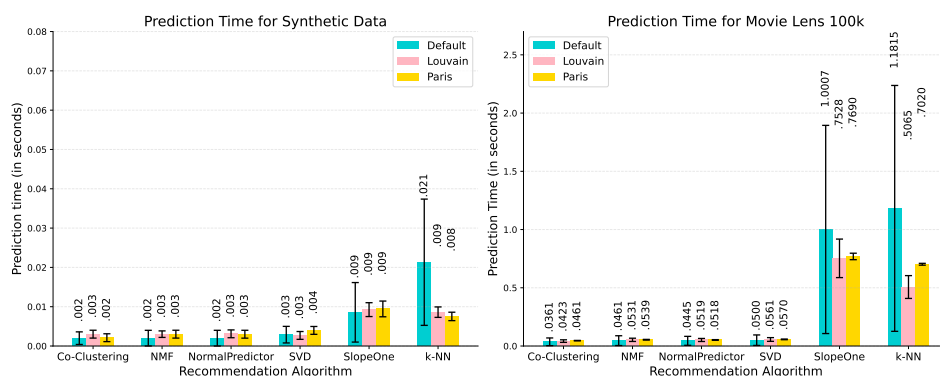


Figure 5: Prediction time considering various datasets, community detection approaches and recommendation algorithms.

computational cost of the community-based recommendation approach was also analyzed. Two aspects were considered: *i*) training time, and *ii*) prediction time. Figure 4 and Figure 5 display the training and prediction time for each algorithm in different scenarios for both datasets. The bar heights correspond to the median execution times, while the error bands depict the uncertainty associated with these median values, calculated using the Median Absolute Deviation (MAD).

While the results for synthetic data indicated a slight increase in computational cost, the use of detectors notably reduced training time for all algorithms in real data scenarios and decreased prediction time for neighborhood-based algorithms.

In summary, our proposed solution approach demonstrates promise in improving recommendation algorithm performance, particularly in datasets with highly connected networks, especially when employing neighborhood-based algorithms. Additionally, even in datasets with fewer disconnected users, the approach maintains similar performance to the default method while significantly reducing computational costs. These findings highlight the approach’s effectiveness in recommendation systems, enhancing performance while optimizing computational resources.

## 5 CONCLUSIONS

In this paper, we introduced and evaluated a community-based approach to enhance the effectiveness and efficiency of recommendation algorithms. Our approach yielded promising results, particularly in data scenarios with clear network structures, consistently outperforming the default method with lower RMSE values. The synergy between community detectors and neighborhood-based algorithms further highlighted our approach’s advantages. Even in datasets lacking distinct clusters, our approach

maintained comparable performance to the default method while significantly reducing computational costs. These findings offer valuable insights for recommendation system researchers and practitioners, showcasing the potential of community-based methods in improving accuracy and computational efficiency.

Future development of our approach should involve exploring whether other algorithms or parameter combinations, both for community detection and recommendation, can achieve similar or superior enhancements within the same pipeline. Additionally, further investigation using different datasets, including real datasets beyond Movie Lens 100k and synthetic data with varying characteristics, can expand our understanding of how our approach handles diverse community structures.

## ACKNOWLEDGEMENTS

Douglas O. Cardoso acknowledges the financial support by the Foundation for Science and Technology (Fundação para a Ciência e a Tecnologia, FCT) through grant UIDB/05567/2020, and by the European Social Fund and programs Centro 2020 and Portugal 2020 through project CENTRO-04-3559-FSE-000158.

## REFERENCES

- Adomavicius, G. and Tuzhilin, A. (2005). 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*, 17(6):734–749. 1
- Bhaskaran, S., Hariharan, S., Veeramani, M., Bharathiraja, N., Pradeepa, K., and Marappan, R. (2022). Recommendation system using inference-

- based graph learning–modeling and analysis. In *2022 2nd International Conference on Innovative Sustainable Computational Technologies (CISCT)*, pages 1–5. IEEE. 1
- Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008. 3, 5
- Bourhim, S., Benhiba, L., and Idrissi, M. J. (2018). Towards a novel graph-based collaborative filtering approach for recommendation systems. In *Proceedings of the 12th International Conference on Intelligent Systems: Theories and Applications*, pages 1–6. 1, 2
- Bourhim, S., Benhiba, L., and Idrissi, M. J. (2019). Investigating algorithmic variations of an rs graph-based collaborative filtering approach. In *Proceedings of the ArabWIC 6th Annual International Conference Research Track*, pages 1–6. 2
- Burke, R. (2000). Knowledge-based recommender systems. *Encyclopedia of library and information systems*, 69(Supplement 32):175–186. 1
- Cao, K. Y., Liu, Y., and Zhang, H. X. (2020). Improving the cold start problem in music recommender systems. In *Journal of Physics: Conference Series*, volume 1651, page 012067. IOP Publishing. 2
- Castells, M. and Cardoso, G. (1996). *The network society*, volume 469. oxford: blackwell. 1
- Choudhary, C., Singh, I., and Kumar, M. (2023). Community detection algorithms for recommendation systems: techniques and metrics. *Computing*, 105(2):417–453. 3
- Çiftçi, O., Tenekeci, S., et al. (2021). Artist recommendation based on association rule mining and community detection. In *Proceedings of the 13th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management-KDIR*. SCITEPRESS. 2
- Dias, P. (2014). From ‘infoxiation’ to ‘infosaturation’: a theoretical overview of the cognitive and social effects of digital immersion. *ambitos*, 24. 1
- Girvan, M. and Newman, M. E. (2002). Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826. 3
- Gorripati, S. K. and Vatsavayi, V. K. (2017). Community-based collaborative filtering to alleviate the cold-start and sparsity problems. *International Journal of Applied Engineering Research*, 12(15):5022–5030. 1, 2
- Guo, G. (2013). Improving the performance of recommender systems by alleviating the data sparsity and cold start problems. In *Twenty-Third International Joint Conference on Artificial Intelligence*. 1
- Hagberg, A., Swart, P., and S Chult, D. (2008). Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States). 5
- Hug, N. (2020). Surprise: A python library for recommender systems. *Journal of Open Source Software*, 5(52):2174. 5
- Keerthi Gorripati, S. and Angadi, A. (2018). Visual based fashion clothes recommendation with convolutional neural networks. *International Journal of Information Systems & Management Science*, 1(1). 1
- Kherad, M. and Bidgoly, A. J. (2022). Recommendation system using a deep learning and graph analysis approach. *Computational Intelligence*, 38(5):1859–1883. 2
- Panteli, M., Piscopo, A., Harland, A., Tutchter, J., and Moss, F. M. (2019). Recommendation systems for news articles at the bbc. In *INRA@ RecSys*, pages 44–52. 1
- Patra, B. K., Launonen, R., Ollikainen, V., and Nandi, S. (2014). Exploiting bhattacharyya similarity measure to diminish user cold-start problem in sparse data. In *Discovery Science: 17th International Conference, DS 2014, Bled, Slovenia, October 8-10, 2014. Proceedings 17*, pages 252–263. Springer. 2
- Paul, P. and Aithal, P. (2018). Digital society: Its foundation and towards an interdisciplinary field. In *Proceedings of National Conference on Advances in Information Technology, Management, Social Sciences and Education*, pages 1–6. 1
- Pazzani, M. J. and Billsus, D. (2007). Content-based recommendation systems. In *The adaptive web*, pages 325–341. Springer. 1
- Pham, M. C., Cao, Y., Klamma, R., and Jarke, M. (2011). A clustering approach for collaborative filtering recommendation using social network analysis. *J. Univers. Comput. Sci.*, 17(4):583–604. 2
- Raghavan, U. N., Albert, R., and Kumara, S. (2007). Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76(3):036106. 3
- Rich, E. (1979). User modeling via stereotypes. *Cognitive science*, 3(4):329–354. 1
- Roser, M., Ritchie, H., and Ortiz-Ospina, E. (2015). Internet. *Our World in Data*. <https://ourworldindata.org/internet>. 1
- Rosvall, M. and Bergstrom, C. T. (2008). Maps of random walks on complex networks reveal community structure. *Proceedings of the national academy of sciences*, 105(4):1118–1123. 3
- Sanderson, M. and Croft, W. B. (2012). The history of information retrieval research. *Proceedings of the IEEE*, 100(Special Centennial Issue):1444–1451. 1
- Schafer, J. B., Frankowski, D., Herlocker, J., and Sen, S. (2007). Collaborative filtering recommender systems. In *The adaptive web*, pages 291–324. Springer. 1
- Xie, J., Kelley, S., and Szymanski, B. K. (2013). Overlapping community detection in networks: The state-of-the-art and comparative study. *Acm computing surveys (csur)*, 45(4):1–35. 5
- Xie, J. and Szymanski, B. K. (2013). Labelrank: A stabilized label propagation algorithm for community detection in networks. In *2013 IEEE 2nd Network Science Workshop (NSW)*, pages 138–143. IEEE. 1