Utilization of Clustering Techniques and Markov Chains for Long-Tail Item Recommendation Systems

Diogo Vinícius de Sousa Silva[1,2][a], Davi Silva da Cruz[1][b], Diego Corrêa da Silva[2][c], João Paulo Dias de Almeida[2][d] and Frederico Araújo Durão[2][e]

[1] *Federal Institute of Maranhão - IFMA, Coelho Neto, Brazil*

[2] *Department of Computing, Federal University of Bahia - UFBA, Salvador, Brazil*

diogo.silva@ifma.edu.br, davi.cruz@acad.ifma.edu.br, {joao.dias, diego.correa, fdurao}@ufba.br

Keywords: Recommender Systems, Clustering, Markov Chains, Assessment Questionnaire, Long-Tail.

Abstract: The primary goal of this paper is to develop recommendation models that guide users to niche but highly relevant items in the long tail. Two major clustering techniques and representing matrices through graphs are explored for this. The first technique adopts Markov chains to calculate similarities of the nodes of a user-item graph. The second technique applies clustering to the set of items in a dataset. The results show that it is possible to improve the accuracy of the recommendations even by focusing on less popular items, in this case, niche products that form the long tail. The recall in some cases improved by about 27.9%, while the popularity of recommended items has declined. In addition, the recommendations to contain more diversified items indicate better exploitation of the long tail. Finally, an online experiment was conducted using an evaluation questionnaire with the employees of the HomeCenter store, providing the dataset. The aim is to analyze the performance of the proposed algorithms directly with the users. The results showed that the evaluators preferred the proposed algorithms, demonstrating the proposed approaches' effectiveness.

1 INTRODUCTION

Recommendation strategies are used in various contexts to bring potential users closer to products with a high probability of interest. In e-commerce, for example, the global market is expected to be worth 6.3 trillion dollars by the end of 2024 - up from 5.8 trillion dollars in 2023, (Snyder, 2024). However, a Recommender System (RS) can negatively impact the diversity of items recommended to users, leading to most recommendations being based on a small diversity of products that have significant success among other system users. Since most users show interest in a particular item, there is a high probability that a new user will also be interested in that item. Following this logic, it is natural for less popular items to be recommended less and consumed less by users (Bobadilla et al., 2013).

Considering the importance of niche groups, which have particular interests, the issue of long-tail recommendations opens up possibilities for studying techniques that improve the performance of such recommendations, not only in recommendation relevance but also in the other aspects mentioned earlier, such as popularity and diversity. Because long-tail items are unpopular products, the accuracy of these recommendations tends to be lower than for items that are consumed by the majority of users (Qin, 2021). Since long-tail items are less popular, they have fewer ratings from other users and consequently provide less input for calculating a more accurate recommendation. In addition, low popularity can lead to more unsold products, resulting in financial losses and the risk of obsolescence or expiry, especially for items with a limited shelf life, such as specialty foods. In the context of online stores, the term "infinite inventory" has emerged, which refers to the set of products that are little consumed but have loyal customers.

With the constant growth of the area, major technology companies also started to invest resources in developing technologies for RS. Research by companies like Google, Netflix, and Amazon showed that such investments could bring good returns for companies and their customers (Gomez-Uribe and Hunt,

[a] https://orcid.org/0000-0002-5683-5133
[b] https://orcid.org/0009-0008-3965-5764
[c] https://orcid.org/0000-0001-7132-1977
[d] https://orcid.org/0000-0002-6617-6696
[e] https://orcid.org/0000-0002-7766-6666

Silva, D., Silva da Cruz, D., Corrêa da Silva, D., Dias de Almeida, J. and Durão, F.
Utilization of Clustering Techniques and Markov Chains for Long-Tail Item Recommendation Systems.
DOI: 10.5220/0012936700003825
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 20th International Conference on Web Information Systems and Technologies (WEBIST 2024)*, pages 47-58
ISBN: 978-989-758-718-4; ISSN: 2184-3252
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Lda.

2015). New services and business models were consolidated based on the technologies developed, helping companies to grow in the market, specifically e-commerce companies like *Amazon.com*, since recommendations facilitated product sales. At the same time, customers and users of these companies also benefited from the advantages and conveniences that such technologies offered, as the generated recommendations saved time and money through suggestions of relevant items.

According to (Anderson, 2006), companies generally focus their sales on the most popular products because it makes logistics easier. If we imagine a traditional store with a physical structure to receive customers, it's easy to understand that it's cheaper to put the best-selling products on the most apparent shelves. However, with the advent of online stores, the cost of organizing products on shelves is non-existent. In the context of online stores, the term "infinite inventory" has emerged, where featured products can be selected according to each user's preference online. This virtual user may not necessarily have the same preference as most other system users.

Alongside "infinite inventory", the phenomenon of the "long tail" emerges. This term refers to products that are rarely consumed but have their niche clientele, meaning niche items. Typically, these products make up the majority of the store's inventory. This large set of products (in the long tail) is responsible for only a minority of sales. On the other hand, only a few products are responsible for most sales made (Brynjolfsson, 2011). An analogy is Pareto's rule, which states that 80% of consequences come from 20% of causes, meaning 80% of sales would be concentrated in only 20% of the inventory.

This paper investigates the lack of long-tail recommendation approaches that prioritize relevance, diversity, and popularity of recommended items. The approach in this paper prioritizes the discovery of niche items while directing them to their target audience. For this, a hybrid approach based on two techniques is used. The first is clustering with dynamic parameters that adapt according to the dataset used, and the second is a type of Markov chain to calculate the interest distance between the user and an item.

The rest of the paper is organized as follows. Section 2 provides a related work for this paper. Section 3 presents two hybrid approaches to the recommendation in long-tail contexts, focusing on the diversity and relevance of recommended items. In Section 4, the experiments conducted to evaluate the proposed approaches will be presented. Section 5 presents the results of the experimental evaluations of the HTCL and P-HTCL approaches. The assessment of the questionnaire applied to business domain users is shown in Section 6. In Section 7, we conclude this paper by presenting the effectiveness of the proposed techniques and discussing future work.

2 RELATED WORK

2.1 Long Tail Recommendation

(Abdollahpouri et al., 2019) researched the problem of popular items bias being frequently ranked highly in RS. The study focused on finding ways to address this issue by increasing the number of recommended long-tail items. The authors approached the problem from the users' perspective, analyzing how popularity bias causes recommendations to deviate from what the user expects to obtain from the RS. Experimental results show that, in many recommendation algorithms, the recommendations users receive are highly concentrated on popular items, even if a user is interested in long-tail items.

(Qin et al., 2020) summarizes in a survey the progress of research on long-tail item recommendation methods, which began with clustering in 2008 and evolved into methods based on deep learning by 2020. Their work reviews the problem of long-tail item recommendation. It describes the main techniques used in this area, such as clustering, graphs, multi-objective optimization, and deep learning, among others. Additionally, it points out future directions associated with this theme by discussing trends in long-tail item recommendation research. (Qin, 2021) provides an update of the work by (Qin et al., 2020), expanding it with a deeper understanding of existing techniques for long-tail item recommendations. (Wang et al., 2006) present an optimized solution for recommending long-tail items. They make recommendations using graph structures to represent the relationships between users and items. The scores of the items are also compared to determine their similarity. The authors assume that a user should evaluate similar items with similar scores. Our proposal also uses graphs; however, the weighted edges will be used to calculate the path from an item node to a user node, whereas (Wang et al., 2006) uses edges to calculate only the similarity between items.

(Yin et al., 2012) developed four variations of an algorithm for long-tail item recommendations. The basic algorithm of the proposal is the Hitting Time, where users and items are represented in a disjoint, indirect, and bipartite graph. An adjacency matrix is obtained from this graph. The graph's edges are weighted and illustrate the relevance of the connection between a user and an item (i.e., the user's rating for that item). To calculate the proximity of unrated items, the author computes the hitting time using a type of Markov chain called "random walk" (Bolch et al., 1998). The random walk calculates the probability of a user reaching an unrated item. The higher the probability, the lower the hitting time, and therefore, the item should have higher priority in the recommendation. Transition matrices are computed from a probability matrix. The probability matrix is obtained from the adjacency matrix. The study by (Luke et al., 2018) proposes a new recommendation system based on tripartite graphs, which aims to suggest long-tail items. The proposed system, the Extended Tripartite Graph System, improves the performance of existing long tail recommendation approaches, measured by widely used performance metrics: recall and diversity. The study highlights the importance of algorithms such as Hitting Time and collaborative filtering approaches to improve long tail recommendations. The results indicate that the proposed algorithm improved long-tail item recommendations, with improvements in Recall@N and diversity.

2.2 Clusterization in Recommendation Systems

(Pang et al., 2019) proposes an improved algorithm for an RS where not only accuracy but also the coverage of recommendation items are considered. For this, the authors use a weighted similarity measure based on genetic algorithms and achieve good results in the context of long-tail item recommendations. In the work of (Yang et al., 2021), a study has been carried out to correct a common problem in recommendations that use clustering and scalability. To this end, co-clustering techniques and clustering models are proposed to make the clustering latent. A score used for each user-item pair is calculated based on their affinities with the clusters. Thus, using the minimum affinity (between the user and the item) for each cluster, the authors show that the method improves recommendation performance on large-scale datasets with millions of users and items with considerably smaller model sizes. (Yadav et al., 2022) addresses a recommendation model that aims to increase the probability of retrieving unusual and new items in recommendation lists while maintaining user relevance. The proposed methodology, Clus-DR (Cluster-based Diversity Recommendation), uses the individual diversity of users and a pre-trained model to generate diverse recommendations. Instead of relying on a reclassification approach, the model uses different clustering techniques to group users with similar diversities. Experimental results with data sets from various domains show that the proposed methodology maintains an acceptable level of accuracy.

The clustering works mentioned above are based on user clustering, which is used to apply a second technique and generate recommendations. Generally, the focus is on improving the accuracy of recommendations or system performance. Our proposal differs in its focus on clustering, which is not performed on the dataset users but rather on the items. Additionally, our focus is not on increasing scalability and performance, as in some of the works presented above, but on suggesting less popular items with more diversity and accuracy.

2.3 Techniques for Long Tail Recommendations

(Lin et al., 2022) propose a method that uses graphs and neural networks to build dynamic and static representations for social recommendation. To do this, they consider dynamic and static representations of users and items and incorporate their relational influence to model the user's interest in a given item. The work of (Abdelkhalek et al., 2022) presents a hybrid proposal for collaborative filtering using belief function theory (BFT), where from the cluster of each item, the K neighboring items are selected as similar and contribute to the recommendation. The overall similarity between the neighbors is chosen for future user recommendations. (Sreepada and Patra, 2021) propose two approaches inspired by econophysics, based on selective injection of ratings into long-tail items using existing rating information. The data is then used to provide recommendations. Test results with real-world data show that the proposed approaches outperform existing techniques in mitigating the long-tail effect and show no significant drop in accuracy.

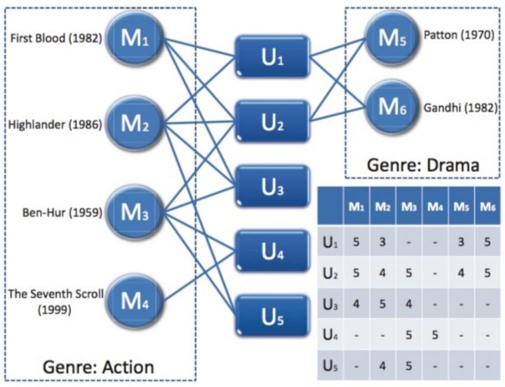
The technique employed in most of the work presented above is Collaborative Filtering. However, as long-tail items receive minimal ratings from users, the tendency is for these items to be inadequately recommended to users. This problem can be mitigated by identifying the long-tail items and incorporating them into the recommendations generated by Collaborative Filtering. Therefore, our work emphasizes these characteristics. We recognize that recommending long-tail items involves identifying them and relevant niche items and directing them to the appropriate users.

3 HYBRID PROPOSALS FOR LONG TAIL RECOMMENDATION

This section will present two approaches that combine strategies aimed at solving the recommendation problem in long-tail contexts A recommendation strategy focused on the long tail should direct suggestions toward items with high diversity and low popularity yet highly relevant to the user. To address these characteristics and based on the analyses and investigations conducted, two hybrid RS approaches were developed in this paper using the Hitting Time algorithm as a basis. The first approach is called Hitting Time Clustered (HTCL), and the second is Personalized - Hitting Time Clustered (P-HTCL), with the latter being an evolution of the former.

3.1 Hitting Time Clustered

In the Hitting Time algorithm (Yin et al., 2012), the set of users U and items M are represented in a bipartite graph, which has its corresponding adjacency matrix. In Figure 1, we have an example representation of the graph with nodes that can be items or users. The edges of these nodes have a weight that reflects the user's score for the item to which the edge connects it. When there is no edge connecting two nodes, it means that the weight is 0 (zero). For example, user node U_1 is connected to item node M_6. In the matrix in Figure 1, we can see that the weight of this edge is 5, meaning that this user's rating for item M_6 is 5. The weight of an edge is represented by $a(i,j)$ with some predefined settings. The variables i and j represent the nodes of the graph in a matrix $A = (a(i,j)_{i,\,j \in V})$. The variable V represents the set of graph vertices.



	M₁	M₂	M₃	M₄	M₅	M₆
U₁	5	3	-	-	3	5
U₂	5	4	5	-	4	5
U₃	4	5	4	-	-	-
U₄	-	-	5	5	-	-
U₅	-	4	5	-	-	-

Figure 1: Representation of users and items through a bipartite graph and its respective adjacency matrix (Yin et al., 2012).

To calculate the proximity of two nodes in the graph, the algorithm uses a type of Markov chain called Random Walk (Bolch et al., 1998). A random walk has a current state (a node in the graph), and with each time step, the state changes, i.e., visiting other nodes in the graph. In Hitting Time, this path is guided by the edge weights, and the random walk terminates its journey when it reaches the destination node. The algorithm relies on a probability matrix calculated from the adjacency matrix. The formula 1 shows how the edge weights are determined:

$$p_{i,j} = P(s(t+1) = j|s(t) = i) = \frac{a(i,j)}{d_i}, \quad (1)$$

where d_i is

$$d_i = \sum_{j=1}^{n} a(i,j). \quad (2)$$

The first proposed approach, HTCL (de Sousa Silva et al., 2020), comprises two distinct yet complementary solutions. This approach falls under the hybridization type called *castata* (Burke, 2002). Next, we present the combination of the Hitting Time algorithm with a clustering technique, optimizing it for long-tail item recommendation.

With the Hitting Time algorithm being used to generate recommendations, we apply an extension to enhance the targeting towards long-tail items. We divide the set of items into short-head and long-tail items. At this point, we used Pareto's rule (Yamashita et al., 2015) as a parameter to separate the more popular items from the less popular ones. All long-tail items are clustered, considering the average score of each item.

The average ratings of long-tail items are taken into account. Those with higher scores will have higher priority in recommendation and carry more weight. These items impact the decrease in the value of the Hitting Time. On the other hand, items with lower scores are grouped into clusters that will be used to weigh the value of the Hitting Time, making it slightly higher. The score of an item is the average of the ratings given by all users who rated it and will be represented in this work by the letter S. Since the possible rating values range from 1 to 5, there will be 4 clusters, one for each score range. The similarity of items within the cluster is calculated based on their score, as shown in the first column of Table 1.

3.2 Personalized-Hitting Time Clustered

The approach of this algorithm involves calculating dynamically the optimal values to be used as Adjust-

Table 1: Clustering of dataset items about mean ratings (score) and its respective adjustment factor(AF).

Score	ClusterFit	*AF (%)*
$1 \leq \underline{S} < 2$	A	+20
$2 \leq \underline{S} < 3$	B	+10
$3 \leq \underline{S} < 4$	C	-10
$4 \leq \underline{S} \leq 5$	D	-20

ment Factor (AF). The algorithm selects a set of ideal values based on some dynamic tests. Subsequently, the set of AF values that achieves the best results in recommendations for long-tail items is chosen. Thus, depending on the dataset and user evaluations, this same model can use different values for the AF of clusters. That is, there will be customization of the AF for each domain, hence the name of the approach: Personalized Hitting Time Clustered (P-HTCL).

To find the ideal values for the AF, the algorithm evaluates several combinations and compares them with each other. As the intention is to explore recommendations for long-tail items, the comparisons are based on the diversity of the recommendations generated with the combination of the AF. Two sequences are considered for the clusters' AF values to test the combinations. Table 2 shows the sequence with the algorithm's values to find the optimal FA value for each of the 4 clusters. The first is an exponential sequence S based on 2, that is, S =[2^0, 2^1, 2^2, 2^3, 2^4, 2^5, 2^6, 2^7....]. Thus, the sequence of values for the AF to be tested in cluster A is S = [1, 2, 4, 8, 16, 32, 64, 128 ...]. The second sequence is applied to cluster B. In this case, the sequence is the natural numbers N = [0, 1, 2, 3, 4, 5, 6, 7, 8 ...]. For the values of clusters C and D, the same sequences are used as clusters B and A, respectively. However, in these clusters, the percentages are negative. Thus, the sequence for cluster C, for example, is N = [0, -1, -2, -3, -4, -5, -6, -7, -8 ...]. Similarly, cluster D is N = [-1, -2, -4, -8, -16, -32, -64, -128 ...].

A diversity measurement is made for each test using the sequences of values shown in Table 2. Each measurement is compared with the results obtained in the previous tests. It is natural for the first tests to reach values closer to the Hitting Time algorithm. This is due to the low influence of the AF, as the AF values are still small. Figure 2 shows that a diversity peak occurs at some point. In this example, the peak happens at test number 4. The following values tend to decrease. The peak represents the ideal value the algorithm will use to generate the recommendations.

Table 2: Sequence of values analyzed by the P-HTCL algorithm in search of the optimal AF set.

Cluster	T1	T2	T3	T4	T5	T6	T7
A	1	2	4	8	16	32	64
B	0	1	2	3	4	5	6
C	0	-1	-2	-3	-4	-5	-6
D	-1	-2	-4	-8	-16	-32	-64

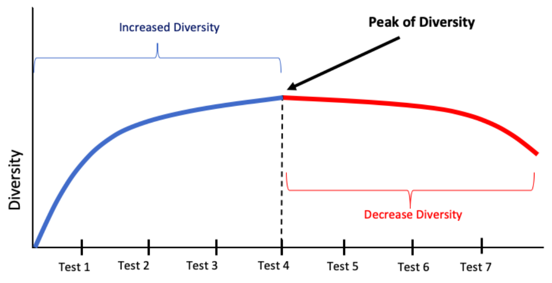


Figure 2: Graph relating the level of diversity of the recommendations to the tests carried out with a set of different values for the AF.

4 EXPERIMENTAL EVALUATION

In this section, the experiments conducted to evaluate the proposed approaches will be presented. The approaches aim to generate recommendations focusing on long-tail items, leading users to a greater diversity of products while maintaining high relevance. The following sections will detail the experimental evaluation setup.

4.1 Methodology

This section shows how the evaluation of the proposed models was configured and the organization of the five experiments. To avoid influencing the results, the averages for each baseline were calculated individually and used the same computational environment.

Several experiments were carried out during this research. However, to avoid excessive length in this paper, we will present only 3 experiments. Additionally, each of these experiments was executed using 3 different metrics. However, for the same reason, we will only show the results with the recall metric here. The third experiment we will present consists of an online experiment where an evaluation questionnaire is administered to users in the business domain. Two datasets were used, one for each offline experiment.

4.2 Baseline

To analyze the effectiveness of the Hitting Time Clustered (HTCL) approach, three comparison methods (baselines) were defined, namely:

- **Hitting Time (HT) -** The Hitting Time algorithm proposed by (Yin et al., 2012);
- **Hitting Time + Clustering All Dataset (HTCA) -** The Hitting Time Algorithm plus clustering is similar to our approach. The difference here is the lack of dataset splitting. In other words, there was no separation of the items between long tail and short head, so the clustering occurred across the entire dataset;
- **Hitting Time + Clustering Short Tail Dataset (HTCS) -** Also similar to the approach used in this work, but the dataset was split inversely. Instead of clustering the items located in the long tail, in this baseline, we cluster only the items present in the short head of the dataset.

To evaluate the performance of the P-HTCL approach, we compare it with Hitting Time (HT) and Hitting Time Clustered (HTCL).

4.3 Datasets

In this section, details of the datasets used in this study, including MovieLens and HomeCenter, will be presented. The MovieLens dataset is widely recognized and used in recommendation research, while the HomeCenter dataset was specially organized for this study based on data from an actual construction store. The analysis of these datasets will provide valuable insights for the evaluation of the proposed approaches.

4.3.1 MovieLens

This dataset was organized by the GroupLens Research group (Harper and Konstan, 2015) and was the first to be used in the experiments in this paper. The MovieLens dataset is used in this research to measure metrics and analyze the results of all baselines in the experiments. It's important to note that this dataset already contains information about user ratings on the items of interest, which includes the relationships between users and items through ratings.

The domain of this dataset is related to movies, with a quantity of 100,000 ratings and a density of 6.3%. Thus, we have a sparse matrix where most users have not rated most movies. However, only users rated at least 20 films from the dataset were selected. This dataset contains approximately 1,683 movies rated by around 943 users. More details about MovieLens are provided in Table 3.

MovieLens also includes metadata such as user age, gender, occupation, and movie category, which were not used in this paper but could be utilized in future work, as presented in Section 7.1.

4.3.2 HomeCenter

This paper's HomeCenter dataset was organized based on data from an actual store. This store is a major retailer in the construction industry, where sales are made physically. The dataset contains approximately 9,138 users and 2,955 purchase items, producing 71,296 sales. Table 3 compares the two datasets presented here.

In addition to being from another domain, this dataset differs from MovieLens in that it does not contain user ratings for items. Therefore, an inference model for ratings was defined based on the quantity of repeated items purchased by customers. In other words, to determine the ratings in this dataset, we considered the number of purchases made by the same customer for a particular product. We then list all the products purchased by each customer, sorting them by the number of times each one was bought. Then, we normalized the quantity of items on a scale of 1 to 5, thus arriving at the same scale used in the MovieLens dataset. Therefore, the more purchases of a particular item, the greater the user's preference. This heuristic was used as a guide for preparing the dataset and inferring the ratings.

Table 3: Comparison between MovieLens and HomeCenter datasets.

	MovieLens	HomeCenter
Number of users	943	9.138
Number of items	1683	2.955
Number of ratings	100.000	0
Domain	Films	Building materials
Sparsity	6,3%	0,7%

4.4 Metrics

This section presents the evaluation metrics used to verify the performance of the proposed approaches in the algorithms. Each dataset was divided into two subsets, one for training and the other for testing and evaluating the recommendations generated. The recall is a fundamental metric for evaluating recommendation systems, indicating the number of items of interest to the user present in the list of recommendations. While popularity helps assess the performance of approaches for less in-demand items.

4.4.1 Recall

The Recall@N metric was used to evaluate the accuracy of the proposed approaches (Yin et al., 2012). Recall is an index that indicates the number of items of interest to the user in the list of recommendations, ranging from 0 to 1, with values closer to 1 indicating a better recommendation.

Calculating Recall@N involves counting how many times an item M appears within the top@N results, as shown in Equation 3:

$$Recall@N = \sum \frac{hit@N}{|L|}, \tag{3}$$

where $|L|$ is the number of test cases. The Recall metric represents the relevance of the recommendations and is calculated by the proportion between the relevant and recommended items and the total number of recommended items.

4.4.2 Diversity

The diversity metric was used to assess the distribution of recommended long-tail items and reduce the influence of popular items on recommendations. With high diversity, less popular items are expected to be recommended to users, allowing the discovery of new items in the long tail. This metric determines the top@N items recommended to a specific group of users. To calculate diversity, we check how many repeated items appear once and then calculate the ratio to the total, according to the equation 4:

$$Diversity = \frac{|UI_u \in I|}{|U|top@N} \tag{4}$$

where I_u is the set of unique items recommended for all users, the element I is the data set, U represents the set of users, and top@N is the recommended number of items for each user (N represents the number of items that the algorithm will return as a recommendation to the user, always from the most relevant to the least appropriate).

4.4.3 Popularity

The popularity metric evaluated the number of long-tail and short-tail items recommended to users. Analyzing the popularity of recommended items in conjunction with other metrics provides a more detailed analysis of the performance of the recommendation algorithm. This metric calculates the frequency of a given item and is based on the ratio of the number of ratings compared to the other ratings in the dataset. The diversity and popularity metrics can measure the extent to which the recommendation is aimed at long-tail items.

For each user, the average popularity is calculated according to the equation 5:

$$Popularity = \frac{R_u}{\sum |R_d|} \tag{5}$$

where,

$$R_u = \frac{\sum |R_r|}{|U|top@N} \tag{6}$$

The ranking defined for the entire dataset is represented by R_d. In contrast, the set U represents the set of users chosen for calculating popularity and top@N, as well as the number of items suggested for each user in the set U. Equation 6 R_u shows a normalized index using the number of users and the items suggested for each one.

5 RESULTS

Figure 3 shows the evolution of P-HTCL tests compared to HT and HTCL. In all top@N, the P-HTCL approach achieves better results than the other two approaches. The best performance is achieved in top@25 when P-HTCL exceeds HTCL by 148%.

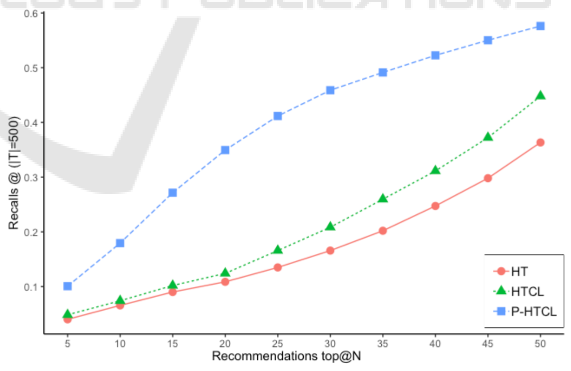


Figure 3: Recall of top@N items from the MovieLens dataset in 500 test cases.

Table 5 shows the results of the recall metrics for each top@N and baseline. As highlighted in bold, note that in the top@25 and top@30, the recall of the P-HTCL approach is twice as good as the HTCL approach.

Figure 4 shows that P-HTCL performed best in all top@N of the three approaches compared. In second place, with a significant distance from P-HTCL,

Table 4: Results of the recall technique from top@5 to top@50 from MovieLens dataset run on all baselines: HT (Hitting Time), HTCL (Hitting Time Clustered) and P-HTCL (Personalized-Hitting Time Clustered).

	Baselines		
	HT	HTCL	P-HTCL
top@05	0,0401	0,0484	0,1007
top@10	0,0656	0,0740	0,1794
top@15	0,0901	0,1019	0,2716
top@20	0,1087	0,1241	0,3495
top@25	0,1350	0,1658	0,4117
top@30	0,1658	0,2085	0,4588
top@35	0,2020	0,2596	0,4913
top@40	0,2473	0,3113	0,5226
top@45	0,2981	0,3722	0,5501
top@50	0,3634	0,4479	0,5762

is HTCL, which appears very close to the last-place HT.

Table 5 shows the absolute values that help us better analyze the results. P-HTCL stands out in top@5 and top@20, performing best compared to the other baselines. At top@5, P-HTCL obtained a recall metric value of 0.40656. Comparing it with the second-best, HTCL, with 0.40010, we see an improvement of 1.61%. Regarding the last-place HT, which only reached 0.39216, P-HTCL surpassed it by 3.67%.

Looking at top@20, P-HTCL reached 0.48186 in the recall metric against HTCL's 0.47502, which is 1.44% better than the second-best approach. Compared to the baseline with the worst performance, HT with 0.47436, P-HTCL obtained an improvement of 1.58%

Table 5: Results of the recall from top@5 to top@50 from HomeCenter dataset run on all baselines: HT (Hitting Time), HTCL (Hitting Time Clustered) and P-HTCL (Personalized-Hitting Time Clustered).

	Baselines		
	HT	HTCL	P-HTCL
top@05	0,39216	0,40010	0,40656
top@10	0,44174	0,44116	0,44766
top@15	0,45948	0,45948	0,46510
top@20	0,47436	0,47502	0,48186
top@25	0,48720	0,48852	0,49312
top@30	0,50098	0,50330	0,50696
top@35	0,51886	0,52066	0,52416
top@40	0,52756	0,53068	0,53332
top@45	0,54530	0,54426	0,54850
top@50	0,56632	0,56788	0,57130

As for the diversity and popularity metrics, the P-HTCL algorithm showed better results than the others, HT and HTCL. Using the MovieLens dataset, the P-HTCL approach stood out at all top@N levels, especially top@10. P-HTCL performed best in diversity at this level, as shown in Table 6, the diversity metric results for each approach.

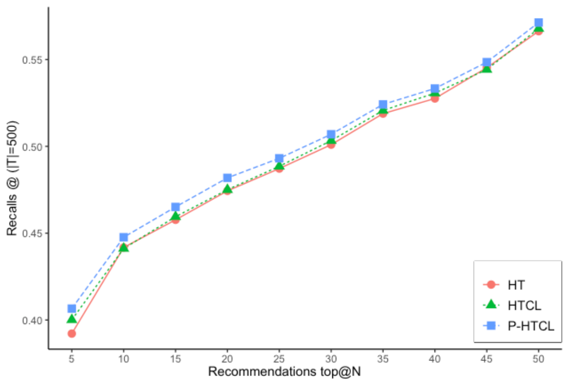


Figure 4: Recall of top@N items from the HomeCenter dataset in 500 test cases.

Table 6: Results of the execution diversity metric in all baselines: HT (Hitting Time), HTCL (Hitting Time Clustered) and P-HTCL (Personalized-Hitting Time Clustered).

	Baselines		
	HT	HTCL	P-HTCL
top@10	0,0535	0,0530	0,0552
top@20	0,0409	0,0408	0,0429
top@30	0,0344	0,0348	0,0369
top@40	0,0305	0,0308	0,0332
top@50	0,0273	0,0277	0,0309

Figure 5 illustrates the evolution of the three approaches, highlighting the superiority of P-HTCL in diversity, especially as the number of recommendations increases.

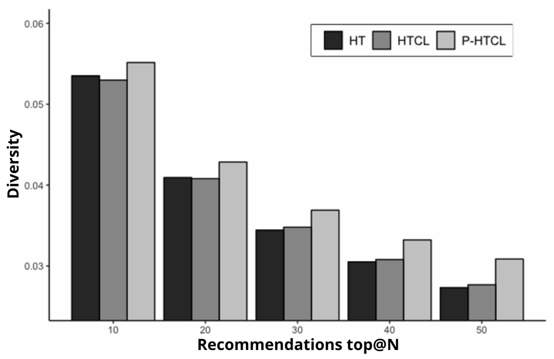


Figure 5: Diversity metric results in Movielens 100k using 200 random users.

In addition, analysis of the results revealed that in the top@50, P-HTCL's diversity was the greatest, which indicates the algorithm's greater ability to offer more varied and less biased recommendations. In terms of popularity, P-HTCL managed to reduce the popularity of recommendations, reaching a value of 1.7095 in top@20, which represents a decrease of 4.16% compared to HTCL, as shown in Table 7.

Table 7: Results of the popularity metric executed in all baselines: HT (HittingTime), HTCL (HittingTimeClustered), and P-HTCL (Personalized-HittingTime Clustered).

	Baselines		
	HT	HTCL	P-HTCL
top@10	1,7546	1,7533	1,7095
top@20	1,2494	1,2429	1,1912
top@30	0,9157	0,9098	0,8737
top@40	0,7250	0,7206	0,6947
top@50	0,6135	0,6075	0,5829

6 complements this analysis by showing the evolution of the popularity metric. P-HTCL not only improves diversity but also focuses on recommending less popular items, which is desirable in long-tail recommendation systems.

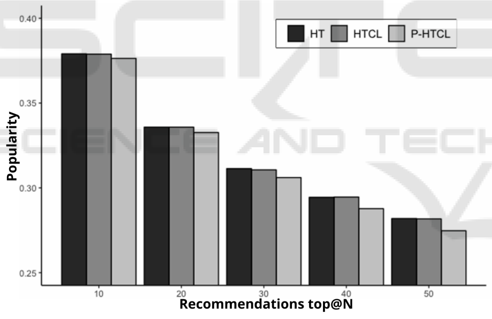


Figure 6: Popularity metric results on Movielens 100k using 500 random users.

Using the HomeCenter dataset, P-HTCL continued to excel in diversity, performing best at all top@N levels, outperforming the other approaches. P-HTCL not only offered diverse recommendations but also managed to reduce the popularity of recommended items. At top@50, the approach achieved a popularity of 0.5829, decreasing by 4.99% compared to HT, which obtained the worst result, as shown in Table 7. It appears that P-HTCL had better results in diversity and popularity, which improves the recommendation of long-tail items. The decrease in popularity shows that recommendations are more effective with niche items. Thus, the P-HTCL approach is a favorable solution for improving the quality of recommendations.

The tests are also conducted 100 times in the ex-

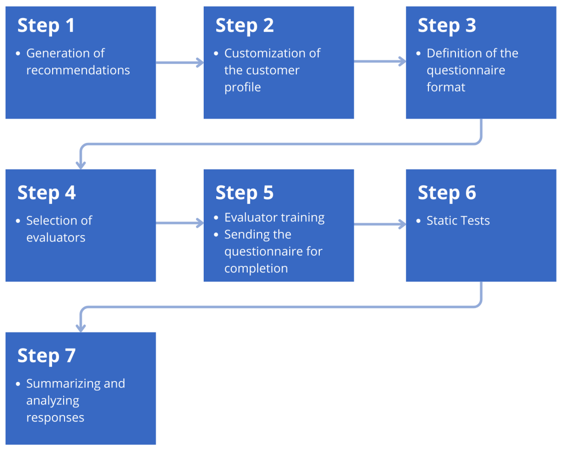

Step 1
• Generation of recommendations
Step 2
• Customization of the customer profile
Step 3
• Definition of the questionnaire format
Step 4
• Selection of evaluators
Step 5
• Evaluator training
• Sending the questionnaire for completion
Step 6
• Static Tests
Step 7
• Summarizing and analyzing responses

Figure 7: Flow of steps for applying the evaluation questionnaire.

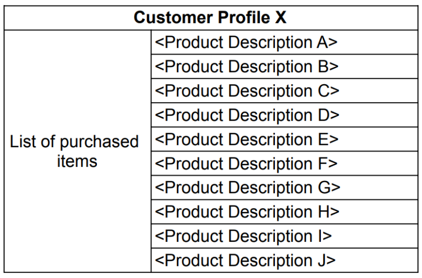
periments to provide statistical confidence. From this data, it is statistically possible to determine if there is a significant difference between the means found. The experiments used the AD (Anderson Darling) test to analyze adherence to normality. After ensuring the means have a normal distribution, we use a parametric test called the T-Test. For the comparison of the averages is considered the p-value < 0.05. All results of the evaluated approaches are compared with the other baselines. In all comparisons, the obtained p-value is less than 0.001. Thus, all results of the metrics presented were statistically different.

6 EVALUATION WITH QUESTIONNAIRE

After performing offline experiments to evaluate the HTCL and P-HTCL approaches, it is also essential to analyze the performance of these algorithms directly with users. In this experiment, the two approaches proposed in this paper (HTCL and P-HTCL) and the HT algorithm were evaluated. An online experiment, through an evaluation questionnaire, was applied to store employees that provide the HomeCenter dataset, i.e., business domain users.

6.1 Questionnaire Execution Steps

To make it easier to understand the procedures carried out during the execution of this experiment, Figure 7 shows a schematic of the steps involved in the questionnaire application process.



Customer Profile X	
List of purchased items	<Product Description A>
	<Product Description B>
	<Product Description C>
	<Product Description D>
	<Product Description E>
	<Product Description F>
	<Product Description G>
	<Product Description H>
	<Product Description I>
	<Product Description J>

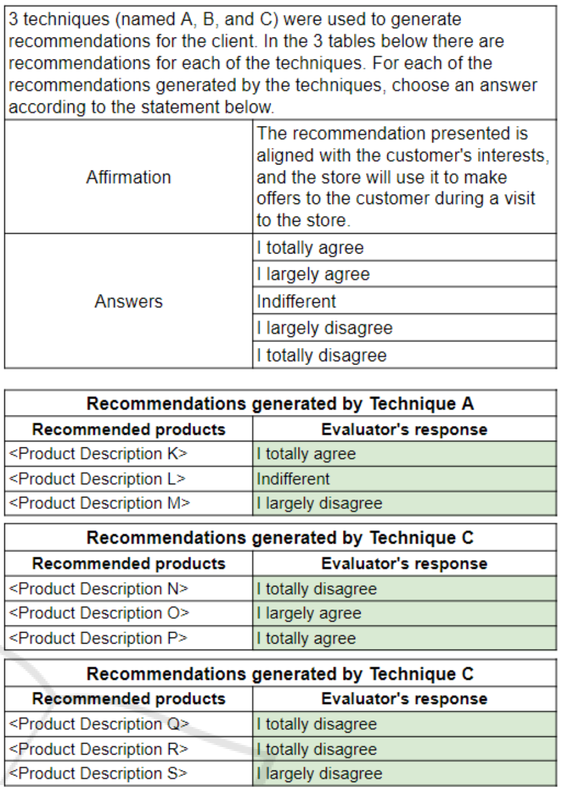
Figure 8: The first part of the evaluation questionnaire presents the client's profile.

6.2 Questionnaire Configuration

The questionnaire consists of a spreadsheet containing questions about the recommendations generated for the customers selected in the experiment. To make it easier to understand, we'll divide this spreadsheet into three sections and explain each. Figure 8 shows the first section of the spreadsheet, which includes a list of 10 previously purchased items. To do this, 10 customers who had previously purchased from the store were randomly selected from the HomeCenter dataset; their personal data was anonymized for this experiment. Each of these customers received 9 product recommendations, generated by running the algorithms (HT, HTCL, and P-HTCL) for each of the 10 customers, resulting in 3 recommendations per approach.

Figure 9 shows the second part of the questionnaire, in which the evaluators express their opinions on the recommendations generated by each technique evaluated. In the questionnaire, the three techniques compared (HT, HTCL, and P-HTCL) are identified by the letters A, B, and C. At the top of Figure 9, there is a brief explanation instructing the evaluator, followed by a statement that must be answered according to the set of answers listed at the top. Thus, for each recommendation generated by the techniques, the evaluator selects an answer based on the following statement:

The recommendation presented is in line with the customer's interests, and the store would undoubtedly use it to make offers to the customer during a visit. In Figure 9, the places where the evaluators provide their answers are highlighted in green. Exemplary answers have been inserted in these places in the illustration for educational purposes only. The Likert scale (Likert, 1932) was used to parameterize and assign values to the evaluation responses. On this scale, evaluators assign graded (ordered) answers to each evaluated question. The questionnaire was administered to two evaluators, and for each of them, ten customers were randomly selected so that the recommendations generated for them could be analyzed.



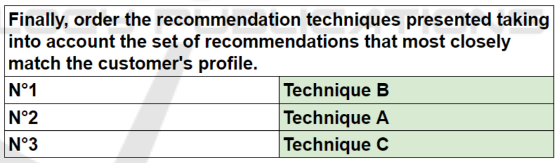
3 techniques (named A, B, and C) were used to generate recommendations for the client. In the 3 tables below there are recommendations for each of the techniques. For each of the recommendations generated by the techniques, choose an answer according to the statement below.	
Affirmation	The recommendation presented is aligned with the customer's interests, and the store will use it to make offers to the customer during a visit to the store.
Answers	I totally agree
	I largely agree
	Indifferent
	I largely disagree
	I totally disagree

Recommendations generated by Technique A	
Recommended products	**Evaluator's response**
<Product Description K>	I totally agree
<Product Description L>	Indifferent
<Product Description M>	I largely disagree

Recommendations generated by Technique C	
Recommended products	**Evaluator's response**
<Product Description N>	I totally disagree
<Product Description O>	I largely agree
<Product Description P>	I totally agree

Recommendations generated by Technique C	
Recommended products	**Evaluator's response**
<Product Description Q>	I totally disagree
<Product Description R>	I totally disagree
<Product Description S>	I largely disagree

Figure 9: The second part of the evaluation questionnaire presents the form with the recommendations generated by each technique.



Finally, order the recommendation techniques presented taking into account the set of recommendations that most closely match the customer's profile.	
N°1	**Technique B**
N°2	**Technique A**
N°3	**Technique C**

Figure 10: The third part of the evaluation questionnaire presents the form with the recommendation techniques ordered by the evaluator.

Finally, based on the recommendations generated for clients, the evaluator establishes an order of performance for the techniques, as illustrated in Figure 10. This creates a ranking in which each method is positioned according to the evaluator's assessment of its effectiveness and suitability. The values highlighted in green in Figure 10 are random examples for illustrative purposes.

The questionnaire was subjected to an evaluation of the degree of agreement and reliability using the general and categorized 5-point Kappa.

6.3 Questionnaire Results

After applying the questionnaire, the evaluators' answers were analyzed, and the scores of the 10 customers were added together to obtain an overall value. The Likert scale (Likert, 1932) was used for this assessment. This assesses the suitability of the recommendations to the customer profile. The P-HTCL technique obtained the best rating with 46.75 points, followed by HTCL with 46.50 and HT with 39.25. Evaluator 1 preferred HTCL, while Evaluator 2 highlighted P-HTCL, the latter being considered the best overall.

Table 8: Summarization of the points obtained in the questionnaire for the evaluator's preference concerning the recommendation generated.

	Reviewer 1	Reviewer 2	Total
HT	17,00	22,25	39,25
HTCL	23,75	22,75	46,50
P-HTCL	22,50	24,25	46,75

Table 9: Summarization of the points obtained in the questionnaire for the evaluator's preference for the technique used.

	Reviewer 1	Reviewer 2	Total
HT	1,0	4,0	5,0
HTCL	7,0	3,5	10,5
P-HTCL	7,0	7,5	14,5

Table 9 shows the evaluators' general perception of the techniques, as shown in Figure 10. The evaluator ranks the methods, giving 1.0 points for first place, 0.5 for second, and none for third. P-HTCL led the way with 14.5 points, followed by HTCL with 10.5 and HT with 5.0 points, indicating the evaluators' preference.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we found that the combination of different techniques proved to be effective in addressing the recommendation problem in the long tail, as there was an improvement in recommendations in various aspects, not limited to just one or two evaluation metrics. The results showed that the techniques used exhibit better relevance indices as recommendations become more diverse and less popular, thus directing users to greater diversity and relevance of products. Therefore, the proposed approaches better guide users to niche items in the long tail. The proposed approaches are hybrid strategies that exploit clustering techniques and representation matrices.

The promising results presented possibilities for retail companies to increase their business profits. Since the profit from the sale of long-tail items tends to be higher than short-head items, focusing part of the sales on these products will bring greater financial returns. The lower competition for these products is evidence of this situation. Additionally, customers of niche products are usually more loyal and are more willing to pay a higher price to acquire them, increasing the profit margin for these companies.

7.1 Future Work

In the experiments with the MovieLens dataset, only one variable, the score, was used. In addition to the score, other variables can be considered in the similarity calculation, such as movie category, producer, cast, or even user clustering using profile data such as age and gender, among others. These variables are already present in the MovieLens dataset and can be the subject of new experiments.

Other techniques can be used with a base algorithm to improve recommendations to exploit a dataset's long tail further. Our approach used Hitting Time as the base algorithm, but other algorithms can be experimented with along with various clustering techniques. Another possibility is to use different methods, such as the probabilistic CF algorithm (IRM2), multimodal similarity, and multi-objective evolutionary algorithm (MORS), to name a few, in conjunction with clustering.

ACKNOWLEDGEMENTS

The authors would like to thank FAPESB and CAPES for their financial support. Grant Term: PPF0001/2021. Technical Cooperation Agreement 45/2021 and CAPES – Grant number 001. This material is partially based upon work supported by the FAPESB INCITE PIE0002/2022 grant.

REFERENCES

Abdelkhalek, R., Boukhris, I., and Elouedi, Z. (2022). Towards more trustworthy predictions: A hybrid evidential movie recommender system. *Journal of Universal Computer Science*.

Abdollahpouri, H., Mansoury, M., Burke, R., and Mobasher, B. (2019). The unfairness of popularity bias in recommendation. https://arxiv.org/abs/1907.13286.

Anderson, C. (2006). *The Long Tail: Why the Future of Business Is Selling Less of More*. Hyperion.

Bobadilla, J., Ortega, F., Hernando, A., and Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems*, 46:109–132. https://www.sciencedirect.com/science/article/abs/pii/S0950705113001044.

Bolch, G., Greiner, S., de Meer, H., and Trivedi, K. S. (1998). *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications*. Wiley-Interscience, New York, NY, USA.

Brynjolfsson, E. (2011). Goodbye pareto principle, hello long tail: The effect of search costs on the concentration of product sales. *Management Science*, 57(8).

Burke, R. (2002). Sistemas de recomendação híbridos: Levantamento e experimentos. *User Modeling and User-Adapted Interaction*, 12(4):331–370.

de Sousa Silva, D. V., de Oliveira, A. C., Almeida, F., and Durão, F. A. (2020). Explorando similaridades em grafos com agrupamento para melhorar recomendações de itens de cauda longa. In de Salles Soares Neto, C., editor, *WebMedia '20: Simpósio Brasileiro sobre Multimídia e a Web*, pages 193–200, São Luís, Brasil. ACM.

Gomez-Uribe, C. A. and Hunt, N. (2015). The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems*, 6(4):13:1–13:19.

Harper, F. M. and Konstan, J. A. (2015). The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19.

Likert, R. (1932). A technique for the measurement of attitudes. *Archives of Psychology*.

Lin, J., Chen, S., and Wang, J. (2022). Graph neural networks with dynamic and static representations for social recommendation. In et al., A. B., editor, *Database Systems for Advanced Applications*, pages 264–271, Cham. Springer International Publishing.

Luke, A., Johnson, J., and Ng, Y.-K. (2018). Recommending long-tail items using extended tripartite graphs. In *Proceedings of the 2018 IEEE International Conference on Big Knowledge (ICBK)*, pages 123–130, Singapore.

Pang, J., Guo, J., and Zhang, W. (2019). Using multi-objective optimization to solve the long tail problem in recommender system. In Yang, Q., Zhou, Z.-H., Gong, Z., Zhang, M.-L., and Huang, S.-J., editors, *Advances in Knowledge Discovery and Data Mining*, pages 302–313, Cham. Springer International Publishing.

Qin, J. (2021). A survey of long-tail item recommendation methods. *Wireless Communications and Mobile Computing*. [Online]. Available: https://doi.org/10.1155/2021/7536316.

Qin, J., Zhang, Q., and Wang, B. (2020). Recommendation method with focus on long tail items. *Journal of Computer Applications*, 40(2):454–458.

Snyder, K. (2024). 35 e-commerce statistics of 2024. https://www.forbes.com/advisor/business/ecommerce-statistics/#sources_section. Accessed: 2024-06-06.

Sreepada, R. S. and Patra, B. K. (2021). Enhancing long tail item recommendation in collaborative filtering: An econophysics-inspired approach. *Electronic Commerce Research and Applications*, 49:101089.

Wang, F., Ma, S., Yang, L., and Li, T. (2006). Recommendation on item graphs. In *Proceedings of the Sixth International Conference on Data Mining (ICDM'06)*, pages 1119–1123.

Yadav, N., Pal, S., Singh, A. K., and Singh, K. (2022). Clusdr: Cluster-based pre-trained model for diverse recommendation generation. *Journal of King Saud University - Computer and Information Sciences*, 34(8, Part B):6385–6399.

Yamashita, K., McIntosh, S., Kamei, Y., Hassan, A. E., and Ubayashi, N. (2015). Revisiting the applicability of the pareto principle to core development teams in open source software projects. In *Proceedings of the 14th International Workshop on Principles of Software Evolution (IWPSE 2015)*, pages 46–55, Bergamo, Italy. ACM.

Yang, L., Schnabel, T., Bennett, P. N., and Dumais, S. (2021). Local factor models for large-scale inductive recommendation. In *Fifteenth ACM Conference on Recommender Systems (RecSys '21)*, pages 252–262, New York, NY, USA. Association for Computing Machinery.

Yin, H., Cui, B., Li, J., Yao, J., and Chen, C. (2012). Challenging the long tail recommendation. In *Proc. VLDB Endow.*, volume 5, pages 896–907.