Proceedings of the Federated Conference on Computer Science and Information Systems pp. 417–425 DOI: 10.15439/2020F157 ISSN 2300-5963 ACSIS, Vol. 21

Dynamic Clustering Personalization for Recommending Long Tail Items

Diogo Vinícius de Sousa Silva
Federal University of Bahia - UFBA
Computer Science Departament,
Salvador - BA, Brazil
Federal Institute of Maranhão - IFMA
Bacabal - MA, Brazil
Email: diogovss@ufba.br

Frederico Araújo Durão
Federal University of Bahia - UFBA
Computer Science Departament,
Salvador - BA, Brazil
Email: fdurao@ufba.br

***Abstract*—Recommendation strategies are used in several contexts in order to bring potential users closer to products with a strong probability of interest. When recomendations focus on niche items, they are called recommendations in the long tail. In these cases, they also look for less popular items and try to find your target custumer, niche market. This paper proposes a long tail recommendation approach that prioritizes relevance, diversity and popularity of recommended items. For that, a hybrid approach based on two techniques are used. The first is clustering with dynamic parameters that adapt from according to the dataset used and the second is a type of Markov chains for to calculate the distance of interest of a user to an item of relevance for this user. The results show that the techniques used have a better relevance indexes at the same time more diverse and less popular recommendations.**

I. Introduction

RECOMMENDATION System is a computational model that aims to approximate the objects to consumers with real interests in acquiring them. To find these associations, a specific research area arose over the 1990s to focus on a study on personalized recommendation techniques [1]. These technicians use several types of recommendation algorithms to process the user profile informations, items metadata, among other information, to conduct a specific user to a service, product, or any other item that is interesting.

In the 20th century, the entertainment industry was based heavily on hits products, that is, those that achieved great success facing the public consumer. However, at the end of the 20th century and the beginning of the century 21th, with the help of the Internet, the industry starts to consider the niche culture. With the experience of the newly created e-commerce sites (with electronic commerce), it was realized that the revenue from a large number of products with low sales index can be equal to or even greater than the revenue of successful products, as these last are made available in a smaller amount of options [2]. If before, the focus on niche products represented high costs with the Internet, the costs of making products visible to customers were smaller and smaller. Before, the focus was on placing products on the shelves more consumed and save with the logistics necessary for the control of stock. With the Internet it became possible to offer products and services each more and more specific, without having to demand costs with physical infrastructure of a showcase or shelves. Products are exposed through images, texts, video files, representing just a few more bits in the infrastructure computationally with low cost. With that, what was previously expensive to be exposed and offered to the public, became accessible, increasing the amount of visible products for potential consumers.

With the increasing spread of the Internet, and through computers, smartphones, TVs, tablets, access has been growing and niches are appearing more. Niche culture is what brings us to the term long tail. So there are greater amount of products available to the consumer and with niches more and more specific. Today, with the Internet and a greater amount of content available, there is a tendency for users to choose more specific items and make the tail longer and longer. Considering the importance niche groups, which have very specific interests, the problem of recommendations on the long tail opens possibilities for studying techniques that improve the performance of such recommendations. They are not relevant to recommendation, but also in other aspects such as popularity, diversity and hybridization. Due to the fact that long tail items are low popular, the accuracy of these recommendations tends to be less than the items most consumed by most users. Since the long tail items are less popular, they have a lower amount of ratings from other users and consequently provide less inputs for the calculation more accurate recommendation.

The combination of different techniques is also an opportunity to decrease the recommendation problem in the long tail. When done correct, can help to improve the long tail recommendations in several aspects, not limited to just one or two metrics as a measure of evaluation. For this reason, the search for new strategies and also the combination of different techniques already in existence is an important topic to be studied. The main objective of this article is to propose a recommendation model that guide better users to niche items, located on the long tail. Thus, it leads them to a greater diversity and relevance products at the same time. For this, clustering techniques and represents matrices will be explored.

Thus, an hybrid strategy will be presented. Considering hybrid strategies, a common technique for recommends that the long tail negatively affect the accuracy already obtained by another technical. This fact will also be addressed in this work so that the approach hybrid approach can add good relevance to the recommendations, without affecting business actively recommend the diversity of recommended items.

It is important to highlight that this work does not intend to analyze the performance related to the execution time, nor of latency. The object of study is limited to improving item recommendations by exploring the long tail. Thus, metrics were chosen to specifically assess the relevance of recommendations from long tail items.

The remaining sections of this paper are divided as follows: in Section II we present a overview in relation to the state of the art in research on recommends long tail. Section III shows the related works. Section IV the model proposed in this work is shown. Section V explains the evaluation model used, as well as the results of this evaluation. Section VI we discuss the conclusions and future work.

II. Background

When studying niche culture, [2] was the first to coin the term long tail. The use of this term was influenced by a paper published on a mailing list by [3] in which the author describes social behavior from the rise of the Internet. The term long tail is the name of a part of a supplier's entire set of items. This part is formed by items that have a small sales margin, however they have a large amount, representing well over 50% of all the stock. By representing this entire volume of items in a graph, such as shown in Figure 1, it can be observed that the right-most side extends until reach zero level. Observing this characteristic of the graphic the author realized that the curve resembled the long tail distribution curve, as denominated in Statistics. It is also common to use the terms tail heavy and fat tail to reference this type of curve in the area of Statistics. In marketing the term long tail started do be used by [2] as a reference to the niche market.

According to [2], very popular products are generally quite commercial by several other companies and, so competition for sales are great. As there is a great demand for these products, the price tends to be as low as possible due to competition. With that, the profit rate of these products is quite low. With items with low demand, it is possible to define a higher profit margin, because users interested in purchasing them will be more willing to pay a higher price for the low availability of the product. Another benefit of exploring long tail products is the one-stop effect shopping convenience, in which the user has the facility to find several types of products in just one store. A store that offers tail products and popular products offer extra convenience for your customer, since this find is everything you need in one place.

On the other hand, the term short head has also come to be used to reference the left-most end of the graphic, it is possible to observe the Figure 1. In this part of the graphic a small number of items are represented, but they are only great success, making the graphic head narrower and, due to the great higher sales quantity. Thus, it was agreed to use the term long tail to represent niche products with great diversity and low popularity. The term short head started to make the products that are highly successful (mass market) and less successful degree of diversity, since they represent a very small number. The greater number of items, the greater the tail. It is on this tail that culture is established niche. Figure 1 shows a graphic with two axis. The abscissa axis represent the items present in the stock of a particular service or product provider. The ordinate axis represent the demand (purchase or download) of each.

With the new characteristics of this niche market, the means of publication content has started to have a greater demand for filtering content. Due to the growing amount of content available, it was increasingly more necessary a mechanism that would help people to find their preferred contents. Two features would need to be implemented: i) make everything available and; ii) help the users to find what they want/need. This gap started to be filled with the Recommendation Systems with approaches specific for the niche market, i.e. recommendations on the long tail.

III. Related Work

The phenomenon of the long tail has been studied with the main objective of leveraging sales of products less by recommending these in conjunction with more popular items. [2] studies how the long tail phenomenon can influence the future of Business. The author brings a more focused analysis to marketing and puts in evidence the importance of Recommendation Systems as a long tail exploration. Thus, the author is able to show how the phenomenon long tail can be seen as a way to increase the profits of a company. [4] propose new algorithms based on chains of Markov for recommending long tail items. Graphs are used to represent the relationships between users and items and, based on those relationships, calculate the distances of interests between these entities. [5] and [6] apply the techniques of adaptive clustering and spplitting of clusters, respectively, to improve performance and decrease the error rate. The last work continue the previous one. In them, the authors evolve a clustering model by making it adaptive to the cutoff point in the dataset for long tail items and short head items. In addition, the size and quantity of the clusters are calculated according to the dataset to be applied the proposed approach. [7] use probabilistic collaborative filtering to generate recommendations for items long tail. Through probability calculations based on relevance models of items the system recommends less popular products and achieves a higher degree relevance in the recommendations. [8] evaluate the coldstart problem and propose a method to generate recommendations for new users based on a user model called Contextual Conditional Preferences. In their experiments, the authors evaluate accuracy measures as well as serendipity, novelty and diversity and obtain recommendations in cold start situations as good as in non cold start situations. [9] discuss the use of collective clustering for recommendation

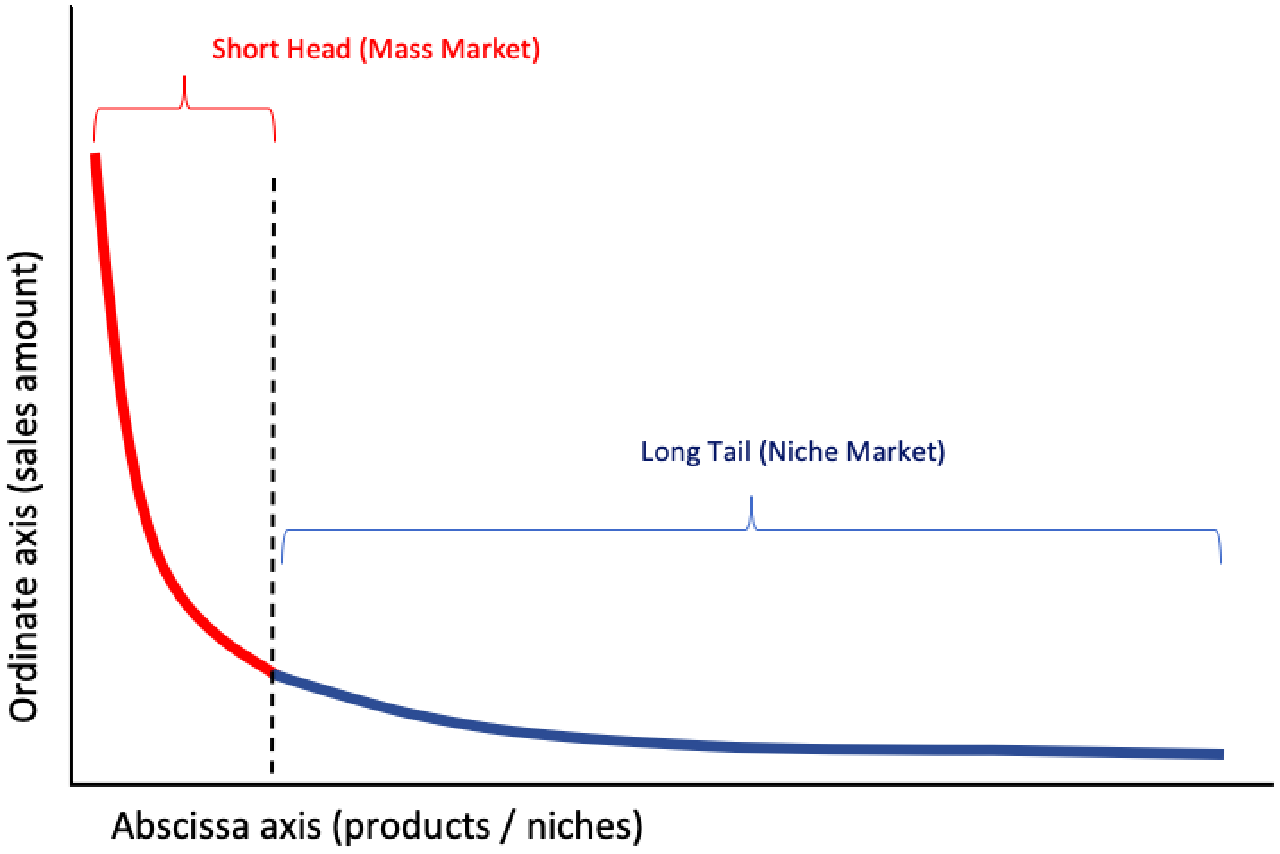

Fig. 1. Representation of the long tail of a hypothetical retailer's inventory relating all products (abscissa axis) to the quantity of sales for each (ordinate axis).

in the context of e-commerce. For this, the authors use data mining combined with RFM techniques. In the work of [10] the authors propose a model hybrid based on the Hitting Time algorithm of the work of [4] and also on the clustering performed in the work of [5]. However, the proposed clustering parameterizes some variables such as number of clusters and the coefficient used to adjust the result obtained in the first technique with Markov chain. With this hybrid approach, the authors perform experiments and presents good results. However, the clustering performed in this work use fixed parameters that may not be the most suitable when used in different datasets.

In most of the works presented above, the technique used is Collaborative Filtering, the most popular for generating recommendations [11]. However, this technique faces a challenge in relation to long tail items. How are items that have a very small amount of classifications by part of the users, the tendency is that these items are little recommended to users. Thus, a relevant item that is in the long tail, has a low chance of being recommended. A simple solution would be identify long tail items, as this task is not costly, and force them to enter the recommendations generated by the Collaborative Filtering. However, this strategy does not guarantee the relevance of recommendations. Recommending long tail items doesn't just mean identifying the items that are present in the long tail. In addition, it is necessary to find relevant items and conduct them to the correct users. At the same time, such recommendations need to be diversified to ensure that each user is actually receiving personalized suggestions. It is also interesting that the recommendation are not popular items, i.e. items that are of interest of few users, and as a consequence generating less obvious recommendations.

IV. Model Description - Personalized-Hitting Time Clustered (P-HTCL)

This section describes the proposed model called P-HTCL (Personalized-Hitting Time Clustered). The model is based on Hitting Time Clustered (HTCL) algorithm [10], which applies clustering and Markov chains to calculate distances of interest between users and items.

A. *Hitting Time Clustered*

For the initial calculation of the distance between users and items, [10] consider the work of [4], in which the dataset with the users' ratings for the items are represented using a graph *user x item*. The edges of the graph are represented by the ratings given by users to the items in the dataset. Representing the graph through an adjacency matrix, the algorithm uses a Markov chain called random walker to traverse through the graph calculating the distance between users and items in the dataset.

Subsequently, [10] apply a clustering technique to generate recommendations more assertive for long tail items. This clustering is applied in the dataset on items classified as long tail. The long tail of the dataset is composed of a set of items that represent the niche market, that is, items with low sales rates, as shown in blue in the graph in Figure 1. In order to classify items in the dataset, the Pareto [12] rule is taken into account. That is, 20% of the most searched items represent 80% of total sales. Thus, the remaining 80% of the items are the least popular and consequently represent the long tail. For the items present in the long tail, the average score of each one is taken into account, i.e. the average of the ratings provided by all users for that item. Using 4 different clusters the items are grouped according to the average score. In our context, a cluster is a set of items grouped according to the average score calculated for each item. In this way, a cluster gathers a subset of items that are present in the long tail. For each of these clusters, a coefficient called the Adjustment Factor (AF) is applied. This factor will conduct the recommendations in order to make them more assertive for long tail items. Thus, the distance between items and users, which was calculated by walking the graph by the random walker, is adjusted according to the following criteria:

- Average Score < 2 — Item allocated to the cluster A and AF = +20%
- Average Score < 3 — Item allocated to the cluster B and AF = +10%
- Average Score < 4 — Item allocated to the cluster C and AF = -10%
- Average Score $<= 5$ — Item allocated to the cluster D and AF = -20%

Under these criteria, there will be a new ordering of the most relevant recommendations according to the new values obtained from the distance between items and users of the dataset.

B. The P-HTCL Algorithm (Personalized Hitting Time Clustered)

The model proposed in the present work performs a calculation to dynamically define the optimal values to be used as AF. The algorithm chooses a set of optimal values based on some dynamic tests. In a sequence, the set of AF values that achieves the best results in the recommendations for long tail items is chosen. Thus, depending on the dataset and the ratings made by users, this same model can use different values for the AF of the clusters. That is, there will be a customization of the AF for each domain, hence the name of the approach: Personalized Hitting Time Clustered (P-HTCL).

In order to find out the optimal values for the AF, the algorithm assess various combinations and compares them to each other. As the intention is to explore recommendations for long tail items, the comparisons is based on the diversity of the recommendations generated with the combination of the AF. For testing the combinations, two sequences are taken into account for the AF values of the clusters. Table I shows the sequence with the values that the algorithm uses to find the optimal AF value for each of the 4 clusters. The first is an exponential sequence S with base 2, i.e. 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 set of 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 ...].

Using the sequences of values shown in Table I, a diversity measurement is performed for each test. Each measurement is compared with the results obtained by the previous tests. It is natural that the first tests achieve values closer to the Hitting Time algorithm. This is due to low influence of the AF, since the AF values are still small. Figure 2 shows that at some point there is a peak of diversity. In this illustration, the peak occurs in test number 4. The following values tend to decrease. The peak found then represents the optimal value that the algorithm will use to generate the recommendations.

C. Personalized Hitting Time Clustered Pseudo-Code

The Algorithm 1 represents the implementation of the P-HTCL. To generate the recommendations a graph with users, items and ratings is required. From there, the algorithm will return the top@N set of items recommended for a specific user. Initially, the dataset is prepared to split long tail items (or niche items) from the rest (also called hit items or short head items). Then, 4 clusters are created to receive the long tail items according to the average score of each one (as shown in lines 2 to 6 of Algorithm 1).

After creating the clusters, it is necessary to define the AF value that will be used in the cluster. In line 7, the variables that will control the tested values are initialized. The tests are performed according to the sequence shown in Table I. The tests are described in lines 8 to 14. In this part of the algorithm, diversity measurements are made with each of the sets of AF values.

The obtained diversity results are compared with previous tests until a pattern is found that indicates a peak of diversity for a given AF value In line 17, the distances from the items to the user are calculated using the Hitting Time algorithm. In lines 18 and 19, the AF value obtained is applied. Consequently, in line 21 there is a need for a new sorting to return the most relevant results according to the Personalized - Hitting Time Clustered approach.

TABLE I
SEQUENCE OF VALUES ANALYZED BY THE ALGORITHM P-HTCL IN SEARCH FOR THE OPTIMAL SET OF THE AF.

Cluster	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10	Test ...
A	1	2	4	8	16	32	64	128	256	512	...
B	0	1	2	3	4	5	6	7	8	9	...
C	0	-1	-2	-3	-4	-5	-6	-7	-8	-9	...
D	-1	-2	-4	-8	-16	-32	-64	-128	-256	-512	...

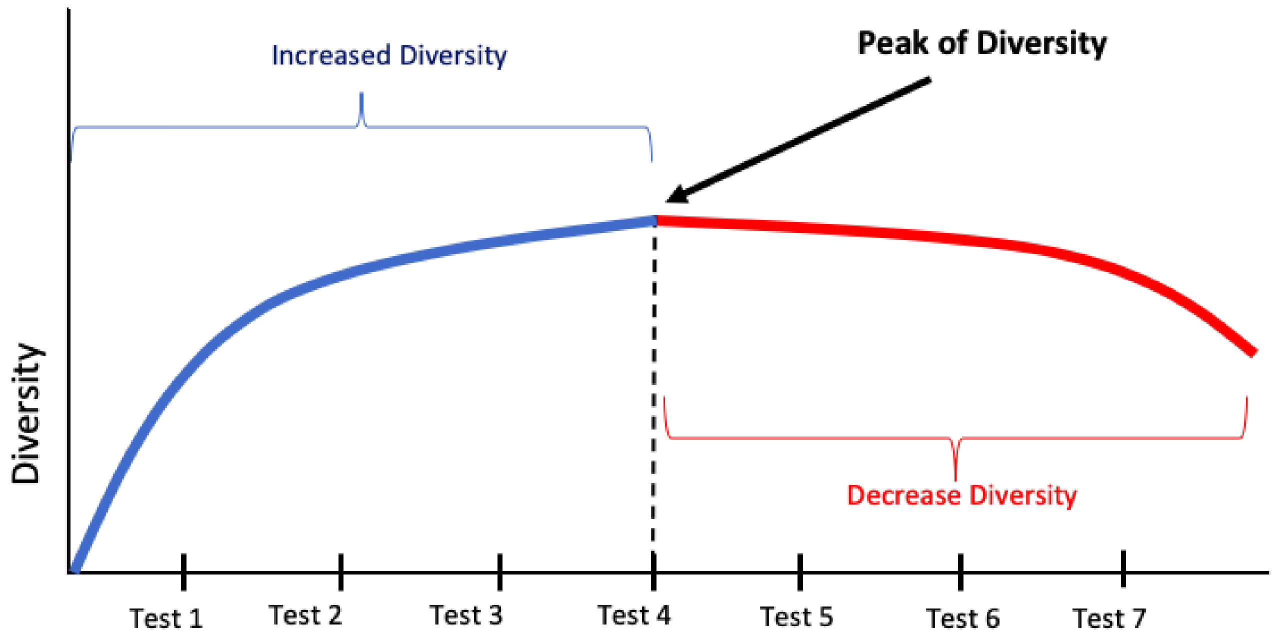

Fig. 2. Graphic that relates the level of diversity of the recommendations with the tests carried out with a set of different values for the AF.

Up to line 14 the algorithm performs an initialization so that the recommendations are computed. This initilization needs to be performed only once. After that, the system will generate recommendations as many times as necessary. Over time, the dataset data will naturally change and a new initialization will be necessary. In this new initialization, it is possible for the AF to change again, since it is calculated dynamically and depends on each dataset. From line 15 our approach will use the initialized data and then the recommendations can be generated. Thus, starting from line 15 this model is possible to run in parallel enironment.

V. Evaluation

This section presents the experimental evaluation of the approach, as well as the comparative results of the proposed model with other baselines.

A. Methodology

We performed offline experiment to evaluate the proposed approach. The dataset is generally divided into two subsets, one for training the recommendation algorithm and the second for testing and evaluating the recommendations generated. The algorithm generate the recommendations based on the set of users who have their evaluation histories already known. The correct ones are computed and then the results of the metrics are obtained. To give more statistical confidence to the experiment, the tests are perform 100 times each. From these data, it is possible to find out if there is a significant difference between the averages found.

To assess of the results of the P-HTCL, we compare it with other two state-of-the-art baselines: i) Hitting Time (HT) - algorithm presented in the work of [4] and ii) Hitting Time Clustered (HTCL) - algorithm presented in the work of [10]. These two algorithms are explained in Section III and Section IV, respectivelly.

The dataset used to measure metrics and analyze the results of all baselines is Movelens [13]. The dataset is a movie domain, with 1,682 titles and 943 users. The total rating is 100,000 with a density of 6.3%, that is, a sparse matrix in which most users have not yet rated most films.

As to the evaluation metrics, according to [14] recall is one of the main metrics used to evaluate Recommendation Systems, and this is used in the evaluation of this work. When we focus on long tail recommendations, it is interesting that the evaluation metrics can reduce the bias generated when recommending short head items. So, to know the performance of a technique for recommending items that have low demand

Algorithm 1: Recommendation using Personalized-Hitting Time Clustered (P-HTCL)

```
Data: Graph with items, users and ratings;
A user and your user-item graph, G(V,E) with
adjacency matrix A;
The amount top@N itens recommendations;
Result:
1  Split item dataset between long tail and short head
     using Pareto's Rule;
2  Create 4 clusters for each rating range;
3  while not at end of itemsLongTailList do
4      Compute item score;
5      Allocate to their respective cluster;
6  end
7  Initialize variable diversity, peak of diversity, sequence;
8  while diversity > peak of diversity do
9      Compute diversity with new adjustment factor;
10     if diversity > peak of diversity then
11         peak of diversity <= diversity;
12     end
13     Select next adjustent factor from sequence;
14 end
15 Select a list itemList with all items unrated by the user;
16 while not at end of itemsList do
17     Compute hitting time from item to user;
18     Apply the respective cluster adjustment factor;
19     Add to map user<personalized hitting time
         clustered, item>;
20 end
21 Sort the map by personalized hitting time clustered
     asc;
22 Return the top@N results;
```

by most users. Thus, the diversity and popularity metrics help evaluate recommendation approaches for long tail items [4], and therefore will also be used in this evaluation. The following metrics were considered:

1) Recall: - This metric represents a measure of relevance of the recommendations. In Equation 1 we have the formal representation of the metric. The calculation is based on the ratio between two numbers. The first number represents the relevant items and at the same time recommended. The second one represents the number of items only recommended. In olher hands, the recall focuses on the number of relevant items that have been recommended correctly, considering all relevant items that may be recommended.

$$Recall = \frac{relevants \cap recommended}{relevants} \tag{1}$$

where the number of relevant items is represented by $relevants$ and the number of recommended items is represented by $recommended$.

2) Diversity: - This metric calculates for a given set of users the top@N items to be recommended. To calculate diversity, we check how many repeated items appear once and then calculate the proportion with the total, as shown in Equation 2.

$$Diversity = \frac{\mid \bigcup I_u \in I \mid}{\mid U \mid \cdot top@N} \tag{2}$$

where I_u is the set of unique items recommended for all users. The I element is the dataset set, U represents the set of users and $top@N$ is the recommended number of items for each user, that is, N represents the the number of items that the algorithm will return as recommendation to the user. For example, $top@5$ means that the algorithm will return 5 ordered recommendations from the most relevant to the least.

3) Popularity: - This metric calculates the frequency of a particular item. It is based on the proportion of the number of ratings compared to the other ratings in the dataset [4]. The calculation is based on the average popularity of the items present in the ranking of each user. For each user the average is calculated to find the final value, according to Equation 3.

$$Popularity = \frac{R_u}{\sum \mid R_d \mid} \tag{3}$$

where

$$R_u = \frac{\sum \mid R_r \mid}{\mid U \mid \cdot top@N} \tag{4}$$

where R_d represents the rating set for the entire dataset, U represents the set of users selected for the popularity calculation and $top@N$ is the number of items recommended for each user belonging to the U set. In Equation 4, R_u normalized rates considering the number of users and items recommended for each one of them.

B. Results

To analyze adherence to normality the experiments used the test AD (Anderson Darling). After ensuring that the means have a normal distribution, we use a parametric test called the Test-T. For the comparison of the averages is considered the p-value < 0.05. All results of the P-HTCL approach are compared with the other baselines. In all comparisons, the result obtained for the p-value is less than 0.001. Thus, all the results of the metrics presented below are statistically different.

1) Recall Results: - Figure 3 illustrates the evolution of P-HTCL tests compared to HT and HTCL. In all top@N the approach proposed in this work achieves better results against the other two approaches. Regarding recall, the best performance is achieved in the top@25, when the P-HTCL exceeds HTCL by 148%.

Table II presents the results of the recall metric in each top@N and in each 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 HTCL approach.

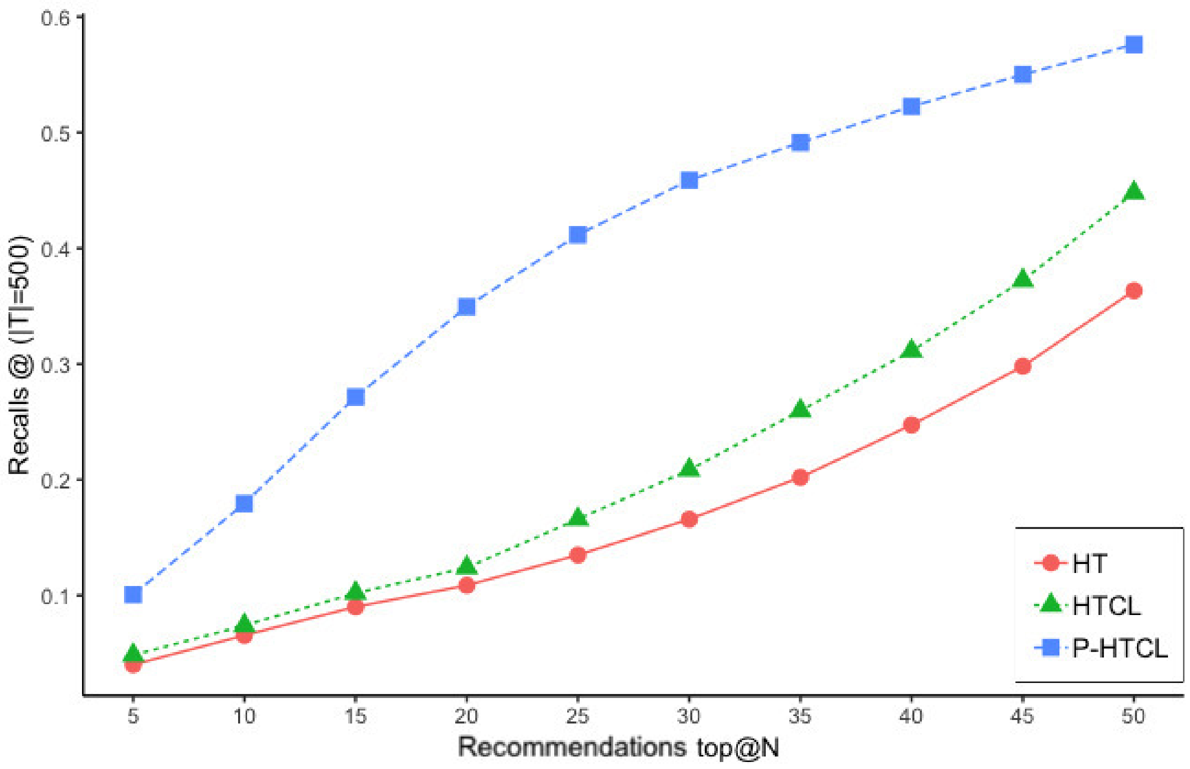

Fig. 3. Recall of the top@N items in 500 test cases.

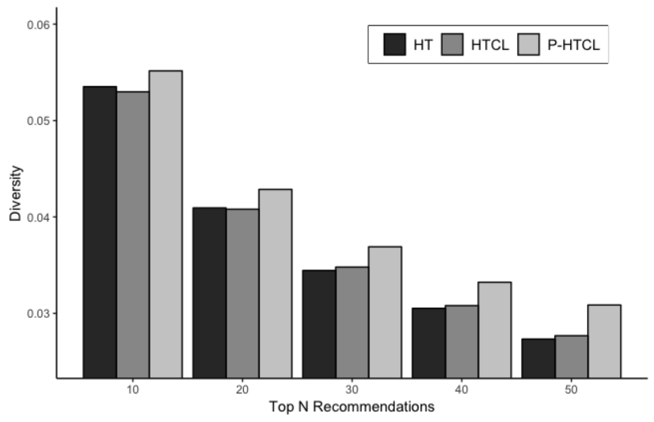

Fig. 4. Results of the diversity metric on Movielens 100k using 200 random users.

2) Diversity Results: - Figure 4 presents the evolution of the three approaches from top@10 recommnedations to top@50. It is possible to observe that from the top@10 the P-HTCL approach always is the best. From the top@20 the difference increases even more, thus showing that it is also the best approach in terms of diversity.

Table III shows the results of the diversity metric of each approach and in all top@N recommendations. Note that the top@50 is the moment when the P-HTCL stands out among the other baselines, in which the diversity of 0.0309 is 11.5% higher than the HTCL, as highlighted in bold in the Table III.

TABLE II
RESULTS OF THE RECALL METRIC FROM TOP@5 TO TOP@50 EXECUTED IN ALL BASELINES: HT (HITTING TIME), HTCL (HITTING TIME CLUSTERED) AND P-HTCL (PERSONALIZED-HITTING TIME CLUSTERED)

	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

TABLE III
RESULTS OF THE DIVERSITY METRIC FROM EXECUTED IN ALL BASELINES: HT (HITTING TIME), HTCL (HITTING TIME CLUSTERED) AND P-HTCL (PERSONALIZED-HITTING TIME CLUSTERED)

	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**

3) Popularity Results: - Figure 5 shows the evolution of the popularity metric of the items recommended in the top@N. Unlike diversity and recall metrics, the popularity is expected

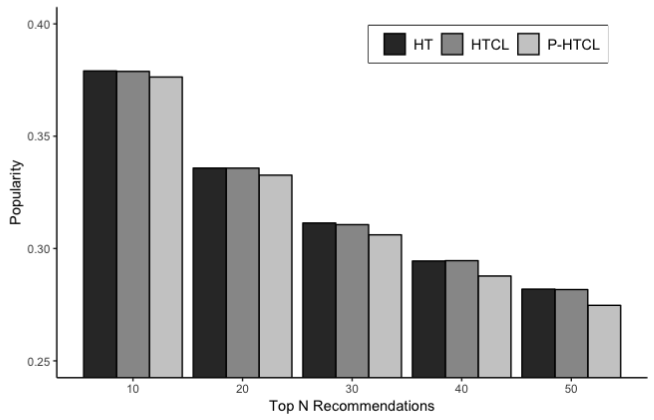

Fig. 5. Results of the popularity metric on Movielens 100k using 500 random users.

TABLE IV
RESULTS OF THE POPULARITY METRIC EXECUTED IN ALL BASELINES: HT (HITTING TIME), HTCL (HITTING TIME CLUSTERED) AND P-HTCL (PERSONALIZED-HITTING TIME CLUSTERED)

	HT	HTCL	P-HTCL
top@10	0,3791	0,3789	0,3763
top@20	0,3358	0,3358	0,3327
top@30	0,3114	0,3106	0,3061
top@40	0,2944	0,2946	0,2878
top@50	0,2819	**0,2817**	**0,2747**

to be as low as possible. In this way, the system recommends less popular items, i.e. more targeted to the niche market. Figure 5 presents that in all the top@N the P-HTCL approach achieves better results than the other baselines.

Table IV presents the average scores. Similarly to the diversity metric, we observe that the greater the top@N the greater the effectiveness of P-HTCL approach. Note that at top@50 we achieve the greatest distance from HTCL with an average of 0.2817, against P-HTCL, which achieves only 0.2747, as highlighted in Table IV.

As a result of the evaluation, improvements have been achieved compared to the HTCL and HT. Recall that the adjustment made was in the automatic configuration of the AF of the clusters used. In HTCL, the 4 clusters are AF: Cluster A = 20; Cluster B = 10; Cluster C = -10; Cluster D = -20. With the customized approach of P-HTCL, various AF are analyzed from measurements of the diversity metric until finding the combination that achieves the best results.

VI. CONCLUSION

This work proposes an hybrid recommendation model to increase diversity, recall and popularity in long tail recommendations. The proposal is an extension of the HTCL model shown in [10]. In this model, the authors propose a graph and cluster based recommendation algorithm. The adjustment factors aim at improving the recommendations on the long tail. These adjustment factors are fixed in the algorithm. In our approach, called P-HTCL, we use a dynamic model to define the value of adjustment factor in runtime. Two rules are used to generate and test variations of the adjustment factors. Then the algorithm tests each one. The first rule is an exponential sequence of order 2 for clusters A and D (the most external), and the second is a numeric sequence natural for clusters B and C (the most internal). In this way, the P-HTCL performs various executions until the results of the diversity metric improve. The objective is to find the maximum point, i.e. the highest peak diversity index.

The tests are performed using Movielens dataset. The results of popularity metric are satisfactory, as it was possible to reduce it by 2,55% percent compared to HTCL on top@50 recomendations. The lower the popularity index of an item, the better for the long tail recommendation, demonstrating that it is a niche item. The opposite happens in the recall and diversity metrics, i.e. the higher the results, the better for the recommendation. As a result, the diversity improved 11,6% over the HTCL method on top@50. The recall improved 148% over the best compared method on top@25. The increase in diversity means that the algorithm recommends a large number of different items. In this way, the algorithm ensures that it does not generate recommendations that are biased in conducting users to the same set of items. In addition, the decrease of the popularity gives us indications that the recommendations are more directly related to long tail items, i.e. the niche items.

The P-HTCL approach has the potential to be used in several contexts, since the P-HTCL does not use any data specific to a particular domain. An great example is in e-commerce. Considering the possibility of increasing sales by exposing more assertively forgotten items, e-commerce companies can benefit from using P-HTCL to recommend items long tail. Video or music streaming companies can also take advantage of our approach, since there is also the positive aspect of exploring the long tail without having to use domain specific data. In addition, it is necessary to carry out more experiments in different domains. The experiments used in our work considered only the Movielens dataset, in which the domain is related to the recommendation of movies. However, for different types of markets, our approach may be applied, such as book recommendation, clothing items, that is, any item that large retailers sell online. In this way, experiments with other datasets are also scheduled as a future work.

One of the limitations of this work is the fixed amount of the number of clusters used, only four so far. We intend to develop an algorithm that will look for the ideal value for the number of clusters to be used. In addition, is important to try other graph based algorithms that focus on improving item recommendations long tail. The work of [15] can be a starting point in this direction. The authors use a tripartite graph that provides more information in the graph structure *user x item*, improving the recommendations.

REFERENCES

[1] G. Adomavicius and A. Tuzhilin, "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*, vol. 17, no. 6, pp. 734–749, June 2005.

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

[3] C. Shirky, “Power laws, weblogs, and inequality,” Economics & Culture, Media & Community, Open Source, 2003, february 08, 2003. [Online]. Available: http://shirky.com/writings/powerlaw_weblog.html

[4] H. Yin, B. Cui, J. Li, J. Yao, and C. Chen, “Challenging the long tail recommendation,” *Proc. VLDB Endow.*, vol. 5, no. 9, pp. 896–907, May 2012. [Online]. Available: http://dx.doi.org/10.14778/2311906.2311916

[5] Y.-J. Park and A. Tuzhilin, “The long tail of recommender systems and how to leverage it,” in *Proceedings of the 2008 ACM Conference on Recommender Systems*, ser. RecSys ’08. New York, NY, USA: ACM, 2008, pp. 11–18. [Online]. Available: http://doi.acm.org/10.1145/1454008.1454012

[6] Y. J. Park, “The adaptive clustering method for the long tail problem of recommender systems,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 8, pp. 1904–1915, Aug 2013.

[7] D. Valcarce, J. Parapar, and A. Barreiro, “Item-based relevance modelling of recommendations for getting rid of long tail products,” *Know.-Based Syst.*, vol. 103, no. C, pp. 41–51, Jul. 2016. [Online]. Available: http://dx.doi.org/10.1016/j.knosys.2016.03.021

[8] A. Karpus, T. di Noia, and K. Goczyła, “Top k recommendations using contextual conditional preferences model,” in *Proceedings of the 2017 Federated Conference on Computer Science and Information Systems*, ser. Annals of Computer Science and Information Systems, M. Ganzha, L. Maciaszek, and M. Paprzycki, Eds., vol. 11. IEEE, 2017, pp. 19–28. [Online]. Available: http://dx.doi.org/10.15439/2017F258

[9] M. Pondel and J. Korczak, “Collective clustering of marketing data-recommendation system upsaily,” in *Proceedings of the 2018 Federated Conference on Computer Science and Information Systems*, ser. Annals of Computer Science and Information Systems, M. Ganzha, L. Maciaszek, and M. Paprzycki, Eds., vol. 15. IEEE, 2018, pp. 801–810. [Online]. Available: http://dx.doi.org/10.15439/2018F217

[10] D. V. Silva and F. A. Durao, “A hybrid approach to recommend long tail items,” in *Anais Estendidos do XXIV Simpósio Brasileiro de Sistemas Multimidia e Web*. Porto Alegre, RS, Brasil: SBC, 2018, pp. 7–12. [Online]. Available: https://sol.sbc.org.br/index.php/webmedia_estendido/article/view/4049

[11] F. Ricci, L. Rokach, and B. Shapira, “Introduction to recommender systems handbook,” in *Recommender Systems Handbook*. Springer, 2011, pp. 1–35.

[12] K. Yamashita, S. McIntosh, Y. Kamei, A. E. Hassan, and N. Ubayashi, “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*, ser. IWPSE 2015. New York, NY, USA: ACM, 2015, pp. 46–55. [Online]. Available: http://doi.acm.org/10.1145/2804360.2804366

[13] F. M. Harper and J. A. Konstan, “The movielens datasets: History and context,” *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, pp. 19:1–19:19, Dec. 2015. [Online]. Available: http://doi.acm.org/10.1145/2827872

[14] A. Gunawardana and G. Shani, “A survey of accuracy evaluation metrics of recommendation tasks,” *J. Mach. Learn. Res.*, vol. 10, pp. 2935–2962, Dec. 2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=1577069.1755883

[15] J. Johnson and Y.-K. Ng, “Enhancing long tail item recommendations using tripartite graphs and markov process,” in *Proceedings of the International Conference on Web Intelligence*, ser. WI ’17. New York, NY, USA: ACM, 2017, pp. 761–768. [Online]. Available: http://doi.acm.org/10.1145/3106426.3106439