2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology

Niche Product Retrieval in Top-N Recommendation

Mi Zhang
School of Computer Science and Informatics, University College Dublin, Ireland.
& School of Computer Science, Fudan University, China.
Email: mi.zhang@ucd.ie

Neil Hurley
School of Computer Science and Informatics, University College Dublin, Ireland.
Email: neil.hurley@ucd.ie

***Abstract*—A challenge for personalised recommender systems is to target products in the *long tail*. That is, to recommend products that the end-user likes, but that are not generally popular. To achieve this goal, in this paper we propose two strategies to identify *relevant* but *niche* products. The first strategy computes an inverse item popularity and applies it during the steps of top-N recommendation. Given a prior probability distribution of relevance based on item popularity, and a user-specific relevance probability, the other strategy uses a number of scores based on distance measures between these two distributions. We emphasize that the problem is to recommend *relevant* items from the user's broader range of tastes. Hence, in evaluation a concentration index is calculated to measure the extent to which the recommendation is spread to the user's niche tastes *in conjunction with* the standard precision metric which measures the overall relevance of the recommended set. The methods are evaluated empirically using the Movielens dataset and show a strong performance in niche item retrieval at the cost of a small reduction in precision.**

***Keywords*-top-N recommendation; long-tail; popularity discount; probability distribution;**

I. Introduction

Many online retail markets are examples of a "blockbuster" industry. While a sales demand curve shows that there are a relatively small number of highly popular products, it also exhibits a tail consisting of a much larger number of products, each of which is liked or bought by much fewer users. This characteristic has been dubbed by Chris Anderson as "The Long Tail" phenomenon [1]. In particular, Anderson finds that specialty items that are not available in traditional brick-and-mortar stores can form a substantial fraction of sales, and argues that the future of business is selling "less of more" [2]. The economics of long tail markets have been further analyzed in [4]. The work provides a theoretical framework and analyzes the motivations that increase the share of niche products both on the supply-side and on the demand-side. Targeting sales of long-tail products is particularly attractive to retailers of digital products since in this case, the cost of storage and delivery of such rarely bought products is not significantly more than that of popular products. Recommender systems have long been considered as one means to help people find and evaluate the many alternatives from the largely expanded offerings in the online world. While it is to be hoped that recommender systems can support the sales of niche products, work such as [11] suggests that recommenders actually "reinforce the blockbuster nature of media", by increasing the concentration of sales among a small sub-set of the product space. One reason for this failing may be that research in recommendation algorithms has largely focused on improving overall system accuracy, while less attention has been paid to developing algorithms that can spread recommendations over a *wider* set of relevant products. In this paper, we focus on strategies to support the recommendation of *niche* and *relevant* products.

As such, we address *top-N recommendation* rather than rating prediction, i.e. the focus is to recommend N products that the systems predicts are likely to be relevant to the end-user, rather than to predict the rating that an end-user might give to any particular product. In the context of top-N recommendation, classification accuracy [13] is more pertinent than predictive accuracy, and classification metrics such as precision and recall should be used to measure system performance. In contrast, much work on recommendation algorithms has been focused on the *rating prediction* problem in recent years, motivated by the Netflix prize. However, note that highly accurate prediction algorithms do *not* necessarily result in algorithms that perform well from a classification perspective, if the items with the highest predicted ratings are selected to form the recommendation set (*predict-and-select*). Our study shows, algorithms that focus specifically on generating a top-N list, such as the item-based kNN algorithm proposed in [8], give a much better performance from the point-of-view of precision, than the predict-and-select strategy. For instance, on the Movielens dataset, using the model-based prediction algorithm proposed in [12], and the experimental methodology described later, we obtain a precision of 0.0125 for predict-and-select in comparison to 0.12 for the kNN algorithm. Hence, in order to support niche recommendation, the strategies proposed in this paper focus on making enhancements to this baseline item-based kNN algorithm and it is referred to as *SR* algorithm in this work.

II. Dataset Analysis

To have a more intuitive sense of the niche item retrieval problem, we perform some analysis on the Movielens

978-0-7695-4191-4/10 $26.00 © 2010 IEEE
DOI 10.1109/WI-IAT.2010.79

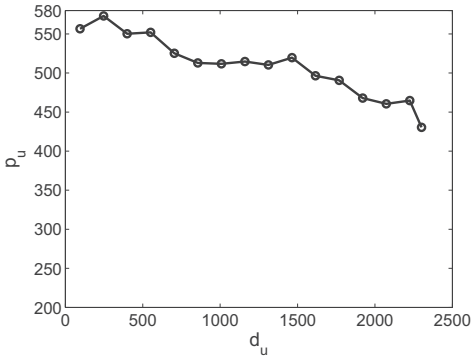

Figure 1. $\bar{p}_u$ against engagement, d_u.

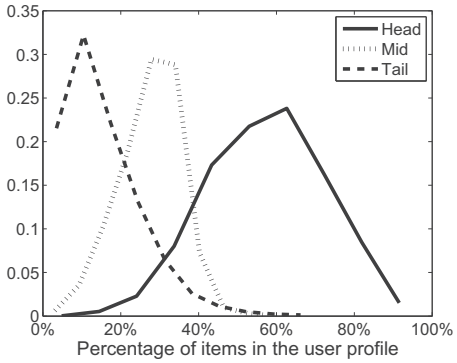

Figure 2. The long-tail distribution of items in P_u.

dataset[1]. The dataset consists of $1,000,000$ ratings that $6,040$ users gave for $3,900$ movies.

We adopt the following notation: Let m be the number of users and n the number of items in the system. For each user u, let $\mathcal{P}_u$ denote the user's profile, i.e. the set of items rated by the user and for each item i, let $\mathcal{U}_i$ denote the set of users that rated item i. Let $l_i = |\mathcal{U}_i|$ be the popularity of item i and $p_i = l_i/m$ be its relative popularity. Let $\bar{p}_u$ be the mean popularity of items in $\mathcal{P}_u$. Let d_u be the number of items rated or consumed by a user u, indicating that user's engagement with the system.

A. Customers' Consumption Behaviour

One might reasonably suspect that heavy users will have a smaller $\bar{p}_u$ than light users (McPhee's "theory of exposure" says that popular products "naturally monopolize" light consumers.). Indeed $\bar{p}_u$ and d_u are negatively correlated (-0.35, with $p < 0.05$ significance t-test) but this correlation is not very strong. Figure 1 depicts the relationship between engagement and mean popularity of consumed items. Users are grouped into 16 bins, according to their engagement levels and each point on the plot corresponds to the mean $\bar{p}_u$ over all users in each bin. We can conclude that heavy users do not have a much stronger intention of consuming niche items than light users.

Furthermore, following a similar approach in [6], we sort the movies according to their popularity and divide them into three parts. Head movies make up the 447 most popular movies that together account for 50% of the transactions in the database; the following $1,175$ movies cover 30% of the total transactions and are placed in the mid part; and the remainder are placed in the tail part. To observe the user preference distribution, we obtain the percentage of items in $\mathcal{P}_u$ that fall in the head, mid and tail respectively. Then the distribution of the percentages for the three parts over all users in the system is depicted in Figure 2, e.g., the solid curve indicates the distribution of the percentage of head items contained in a user profile. It shows that a big proportion of users do consume popular items much more often than niche items. On average, 57% of a user profile consists of head items, 28% of mid items and 15% of tail items. However, it also indicates that if we only focus on recommending head items we are missing more than 40% of the total consumptions in the system. Many users still consume a much bigger proportion of niche items than average, e.g, for 28% of users, more than 20% of their profile is made up of tail items and there are even some users whose profiles contain a majority of tail items.

Another conventional perception is that customers commonly appreciate niche items much less than popular items, i.e. such items receive very low ratings. We examined the mean ratings that each part received. They are 3.82, 3.43 and 3.14 for the head, mid and tail, respectively, which indicates that in general users appreciated niche items only a bit less. In fact, our study shows that 71% of the ratings received by tail items are ≥ 3 and 43% of the ratings received by tail items are ≥ 4. That is, customers may rate the niche items less but not always lower.

Table I
CATEGORY LOCATIONS OF SIMILAR MOVIES (MOVIELENS DATASET).

	Head	Mid	Tail
Head	91.09%	8.91%	0
Mid	41.22%	52.79%	5.99%
Tail	2.78%	25.59%	71.63%

B. Concentration Reinforcement

To investigate how a recommendation algorithm can reinforce the concentration in the dataset, for each movie, we identify the top 20 most similar movies, using the cosine similarity metric. In Table I, the extent to which similarity correlates with movie categories is shown. For instance, Row 1 indicates the case for a "head" movie. Thus, 91% of the time, a movie in the top 20 most similar movies to a head movie is also a head movie. The implication is, in the

[1] http://www.movielens.org

similarity space the closest movies to popular movies are also popular movies and hence a *similarity-based* algorithm is likely to be biased towards recommending popular movies. On the other hand, given a tail movie, its similar movies are mostly located in the tail, too. The conclusion is that it is inevitable that with a similarity-based recommender the already popular items are very likely to be reinforced.

III. Evaluation Metric

It is not enough for a recommendation algorithm to spread recommendations over a wide range of different product types. Rather, it is necessary to find those less generally popular products, which are highly likely to be *relevant* to the active user of the system. This motivates us to evaluate success in niche product recommendation using the *concentration of hits* – that is, the extent to which *relevant* recommendations are concentrated among a sub-set of the product-space. The smaller the concentration, the better the system is at making niche recommendations. This ability is encapsulated in the *concentration index* (CI) and the associated concentration curve. Precision is used to measure the overall quality of the recommendation, as the percentage of the recommended set that contains relevant items.

To compute a *concentration curve* [15] of the hits distribution against popularity, the products are ordered in terms of increasing *popularity* and the cumulative proportion of *hits* is plotted on the y-axis against the cumulative proportion of products on the x-axis, where *a hit* is defined as the recommendation of a product that is known to be liked by the user. A point (x, y) on the curve should be interpreted as $x\%$ of the least popular products account for $y\%$ of all hits. Associated with the concentration curve is the concentration index CI. It may be computed as twice the difference between the area below the diagonal and the area below the concentration curve $c(x)$: $CI \triangleq 1 - 2\int_0^1 c(x)dx$, CI lies in the range $[-1, 1]$. If it is zero, then the hit probability is not correlated with popularity. Negative values imply a bias towards unpopular items, positive values imply a bias towards popular items.

Absolute Concentration and the Gini Index: While the primary focus of the techniques discussed in this paper is to remove the recommendation bias towards popular items, one danger is that the new algorithms will concentrate over a different set of items and that the system will still be limited to recommendations of only a small number of the relevant items, albeit not just popular ones. The absolute hits concentration can be measured using the *Lorenz curve*. Different from the concentration curve, to plot a Lorenz curve the products are ordered in terms of increasing *number of hits* on the x-axis, a point (x, y) on the curve indicates that $x\%$ of the products with *smallest* number of hits, account for $y\%$ of all hits. The Lorenz curve corresponds to the diagonal only when all relevant products have an equal chance of being recommended. The index corresponding to the Lorenz curve is called the *Gini index* and lies in the range $[0, 1]$. The closer the Gini index is to 0, the more the hits are dispersed among a wider range of relevant products. We will see that, as the tendency to recommend popular products is a major cause of the concentrations observed in recommender systems, the algorithms that we propose to remove this bias also succeed in reducing the overall concentration, as measured by the Gini index.

IV. Related Work

There are two opposite opinions towards the problem of whether we should invest in the long tail from the profit driven point-of-view. The representatives of the two sides are Chris Anderson [1] and Anita Elberse [9], respectively. For example, Anderson [2] describes consumers' demand for niche products in an age of infinite-inventory retailers. On the other hand, Elberse [10] has suggested that tail inventory is overrated. Regardless of the importance of tail sales, we believe that niche product recommendation is important to increase *consumer* satisfaction. Strategies that increase the probability of recommending relevant products in the long tail are necessary to support true *personalisation* in the sense that the system targets each users' special needs. That is, in order to provide more *personalized* recommendations, systems need to be able to make recommendations of products that the current user likes, but which are liked by relatively few others. In terms of addressing this issue, the previous work which most closely follows our research is [11]. In this work, the impact of a recommender system on sales diversity is analysed. The somewhat discouraging result from the perspective of the long tail problem is that, in most cases, the authors show that a recommender that bases recommendations on previous sales history tends to *increase sales concentration*. The evaluation of sales diversity is based on an analytical model rather than empirical analysis using real-world data. The model assumes that the effect of a recommender system is to change the probability that a user will purchase a particular product. We take the view that the role of a recommender is not to increase users' intrinsic probability of purchasing any particular product, but rather to make users *aware* of certain products that they have a high probability of purchasing, once they are known to them.

V. Popularity Discount

In this section we present two strategies based on popularity discount. The first applies popularity discount before making the final recommendation. The second takes account of popularity discount in computing the similarity matrix. Both strategies use a notion of *inverse popularity* which is defined for an item i as

$$InvPop(i) = (1 - p_i)^b \tag{1}$$

where p_i is the relative popularity of item i and b is used to amplify the effect of popularity discount, $b = 1$ by default. The inverse popularity of the items is plotted in Figure 3.

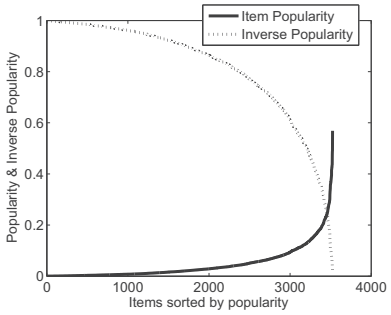

Figure 3. Inverse popularity of the items.

These two strategies perform the popularity discount by modifying the *SR* algorithm. In this algorithm, firstly a *candidate sub-set*, C, of items from which the final recommendation is drawn is identified by selecting the K most similar items to each item in $\mathcal{P}_u$ (removing duplicates). The recommended set is then identified from these, through a more refined measure of similarity that is typically calculated as the aggregate similarity between a candidate item and *all* items in $\mathcal{P}_u$.

A. Inverse Item Popularity Discount (IIP)

For the *IIP* strategy, after the *SR* algorithm has been applied to calculate a similarity value for each candidate item to P_u, these similarity values are multiplied by $InvPop(i)$ and the items are ranked according to the modified value.

B. Utility of User Profile (UUP)

The *UUP* discounting strategy is inspired by the *profile utility* measure proposed in [14]. Originally this was applied in the context of a user-based recommender system where each user profile is assigned a global weight that models the profile's usefulness as a potential *neighbour*. The profile utility of user u is defined as:

$$U(u) = \frac{1}{|\mathcal{P}_u|} \sum_{i \in \mathcal{P}_u} InvPop(i)\,. \tag{2}$$

We use (2) to discount the cosine similarity metric. Writing $s(i,j)$ as the similarity between two items i and j, we define a popularity discounted cosine similarity as:

$$s(i,j) = \sum_{u \in \mathcal{U}_i \cap \mathcal{U}_j} \frac{U(u) r_{u,i}}{\sqrt{\sum_{v \in \mathcal{U}_i} (U(v) r_{v,i})^2}} \frac{U(u) r_{u,j}}{\sqrt{\sum_{v \in \mathcal{U}_j} (U(v) r_{v,j})^2}},$$

where $r_{u,i}$ is the rating given by a user u to an item i.

VI. Probabilistic Scores

The above popularity discount algorithms can be regarded as a simple modification that finds obvious recommendations, and reduce the chance of the obvious ones appearing in each recommendation list before presenting it to users. A possible disadvantage of it is that since each user has had different consumption history in the past the obvious items might be different for them. An alternative would combine what is known about the user's tastes with what is known about the *average* tastes from all the users.

A. Motivation

A key component of *SR* is the similarity measure that provides a heuristic measurement of how suitable an item i is for recommendation to a user u. From the classification perspective, the recommendation problem may be viewed as a decision problem in which the issue is to decide whether an item is relevant to a user, given the similarity. A statistical approach to this problem will base this decision on a statistical model of item relevance. Thus we may try to model the probability of relevance, given the similarity, which we write as $p_L(s)$, where L is the event that an item is liked by the user. On the other hand, a simple *prior* probability of item relevance, independent of any particular user, can be estimated as the relative popularity of each item, p_i. Note that in general high similarity tends to be correlated with high popularity, so simply selecting those items that are most likely according to $p_L(s)$ leads to the observed problem of recommending the most popular items.

Instead, we identify possible niche items as those items for which $p_L(s)$ differs from p_i. Our strategy for niche recommendation can be summarised as firstly, building the above statistical models and then using measures of distance between the distributions to identify niche items. Intuitively, the distance represents the amount that the given user will like the product more than most other users. Unpopular items will often be recommended if the current user is particularly interested in them, while very popular items will be recommended only if the current user is exceptionally interested in them. This approach will discover surprising items that are liked by the users.

We incorporate this approach into the *SR* algorithm *after* selection of the candidate set C and use it as a means of selecting the recommended set R from C. Specifically, when making recommendations to a user, each of the items in the candidate set C is assigned a score D_* with $*$ being one of the criteria described below. They are sorted in descending order according to D_* and the top N items are selected to form the recommendation list.

B. Measuring the Distance between Distributions

The Kullback-Liebler divergence (KLD) [7] or information gain is defined for discrete distributions by

$$KL(p,q) = \sum_{x \in J} p(x) \log \frac{p(x)}{q(x)}.$$

where p and q are two probability distributions, x is one possible outcome in the space of all possible outcomes, J.

In this work, $J = \{0, 1\}$, where 1 represents the outcome of relevance, i.e. the item is liked by the user, and 0

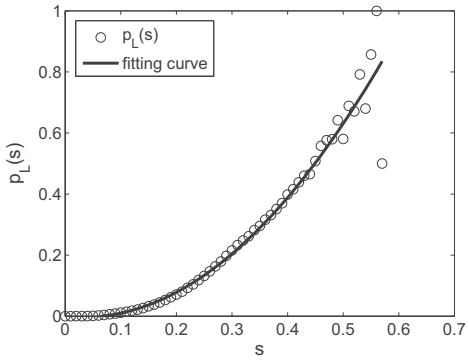

Figure 4. $p_L(s)$ - Probability to be liked against s.

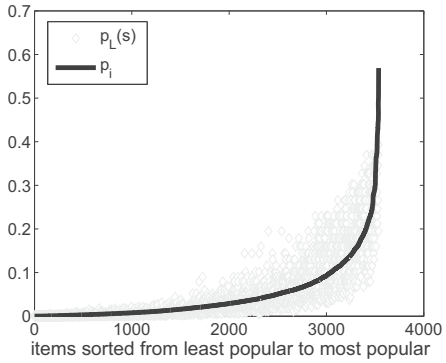

Figure 5. Distributions of $p_L(s)$ and p_i.

represents the outcome of irrelevance. Then, we can compute the difference between posterior and prior distributions as a score, D_{KL} obtained by an item i for the current user u:

$$D_{\mathrm{KL}} = p_L(s) \log \frac{p_L(s)}{p_i} + (1 - p_L(s)) \log \frac{(1 - p_L(s))}{(1 - p_i)}.$$

A user-item pair with a high KLD score corresponds to an item for which the predicted suitability of this item to this specific user is different to their overall likelihood of relevance due to its popularity.

Additionally, we apply another distance metric between two distributions – L1-norm distance [5]:

$$D_{\mathrm{L1}} = \frac{1}{2} \sum_{x \in J} |(p(x) - q(x))|$$

The L1-norm score in our context is :

$$\begin{aligned} D_{\mathrm{L1}} &= \frac{1}{2}(|p_L(s) - p_i| + |p_i - p_L(s)|) \\ &= |p_L(s) - p_i|. \end{aligned}$$

However, since it always holds that $D_{\mathrm{KL}} \geq 0$ and $D_{\mathrm{L1}} \geq 0$ these scores do not distinguish the case that the difference is due to $p_L(s)$ being high when p_i is low, from $p_L(s)$ being low when p_i is high. We therefore propose some other heuristic scores.

C. Other Scores

We assign a score to each candidate item with the following criteria based on the idea that the *value* of an item being recommended is low when it is generally popular:

- Ratio: $D_{\mathrm{R}} = \frac{p_L(s)}{p_L(s)+p_i}$. The ratio score lies in the range $[0, 1]$ and is maximised when $p_L(s) = 1$ and $p_i = 0$.
- Product: $D_{\mathrm{P}} = (1 - p_i)p_L(s)$, where $1 - p_i$ represents the probability that the item is *not* commonly liked. The idea for this criteria is, if an item is not liked by the majority (high $1-p_i$) but liked by the current user (high $p_L(s)$), it can be regarded as the user's niche taste.
- Difference: $D_{\mathrm{d}} = p_L(s) - p_i$. The difference between the two relevance probabilities. This removes the absolute values from the L_1 norm, since we are interested only in the case of high $p_L(s)$ and low p_i.

Moreover, we construct a weighted sum of the above three scores. That is:

- Weighted Arithmetic Mean (WAM):

$$D_{\mathrm{WAM}} = \alpha_1 D_{\mathrm{R}} + \alpha_2 D_{\mathrm{P}} + \alpha_3 D_{\mathrm{d}}$$

where $\sum_{j=1}^{3} \alpha_j = 1$ and $\alpha_j \geq 0$. The parameters α_j are chosen for each user to maximise the score over known relevant items in the user's profile. Given a set of scores $D_{\mathrm{WAM}}(i)$ for each item $i \in P_u$, we choose α_j to minimize

$$\sum_{i \in P_u} (1 - D_{\mathrm{WAM}}(i))^2$$

using a constrained least squares fit.

D. Probability Distributions

We estimate the probability distributions $p_L(s)$ and p_i empirically from a training dataset of ratings provided by a set of $3,000$ randomly chosen users. We calculate the similarity values for all the relevant items in each user profile to obtain $s(u, i)$ as the average similarity of item i to the other items in the profile $\mathcal{P}_u$. We also calculate the similarity for all remaining pairs (u, i).

The entire range of similarity values s is divided into k bins $B = \{B(1), ..., B(k)\}$. Let $c_B = \frac{\max(p_L(s)) - \min(p_L(s))}{k}$, the range for the j^{th} bin is $[(j-1)c_B, jc_B)$. For all users in the dataset, we count the number of times that similarity values for relevant items fall in each bin $B(j)$, and hence the fraction of observed *relevant user-item pairs* that fall in each bin, which is written as $n_L(j)$. In the same manner, we obtain $n_r(j)$, the fraction of *all observed user-item pairs* that fall in each bin. The ratio of these two numbers gives an estimate for $p_L(s)$. Specifically, for the j^{th} bin, the probability that the items with similarity s falling in this bin are actually liked by the user, is calculated by $p_L^j(s) = \frac{n_L(j)}{n_r(j)}$. The resulting values of $p_L^j(s)$ can be represented as a vector $p_L(s)$ of length k. A quadratic curve is fitted to the raw $p_L(s)$ values obtained from the above process, see Figure 4. Note that only the highly rated (with ratings of 4 or 5) items in the users' past transactions are regarded to be relevant.

Note that as defined earlier, p_i is fixed for each item, i.e. it is not user-sensitive. On the other hand, $p_L(s)$ is user-sensitive since it depends on the s value for each user-item pair. To have a more intuitive sense, we display the two probability distributions for one user in Figure 5. It can be seen that the trends of these two probability distributions are similar, but there is a certain distance between them for individual items and for some items that distance can be big. In the experiments, the probability of $p_L(s)$ is trained for each user independently, i.e. the probability of items being liked by a user depends on the similarity distribution from the current user, rather than the similarity distribution drawn from all the users.

VII. Experiments

A few baseline algorithms we applied for recommendation from the candidate set C are listed as follows:

- SR: the N most similar items to P_u are recommended.
- Random recommender (RR): Select N items from C uniformly at random.
- Popular list recommender (PR): recommend the N most popular items in C.

We compare these algorithms against our niche product recommendation strategies.

Moreover, our strategies are dependent on the choice of the item-item similarity metric. We use standard cosine similarity and additionally, we use a reduced dimension similarity metric. Since latent semantic indexing [3] was first proposed, it has long been recoganised that dimension reduction can help to uncover similarities that are otherwise difficult to recognise in the original high-dimensional space. For example, [16] pointed out that we can take the advantage of reduced dimensionality to form better neighborhoods of customers. We carry out an SVD factorisation of A, the $m \times n$ dimensional matrix of ratings for the n items in the dataset. Given $\mathrm{A} = \mathrm{U}\Sigma\mathrm{V}^T$, we take the first l columns of V to form an l-dimensional representation of the items s.t. $m \gg l$. This representation is used to compute item similarities using the cosine metric on the reduced dimension item vectors. Clearly, the choice of dimension l is critical to the success of this method, among all the values between 20 to 200 with interval of 10, $l = 50$ provides the best average performance and is applied in the experimental section. We apply this dimension reduction method (SVD) to both the *SR* and *IIP* algorithms.

We evaluate the algorithms on the Movielens dataset, selecting Y_T as the set of users with non-empty profiles. For each user u, a test set T_u of size $10\% \times |\mathcal{P}_u|$ is selected with equal probability from $\mathcal{P}_u*$ and removed from Y_T, where $\mathcal{P}_u*$ consists of items with ratings of 4 or 5 in $\mathcal{P}_u$. In all experiments the size of the recommendation set R is fixed as $N = 20$. A hit occurs whenever R intersects T_u. The transactions left in Y_T are used for the training of probability distributions. All the experiments are repeated 10 times and the average values are taken. The differences of the results between different algorithms are all statistically significant ($p < 0.05$ with t-test).

A. Results

The concentration curves of popularity discount strategies are depicted in Figure 6. Results are presented for the standard similarity measure and also the reduced dimension measure (SVD). The corresponding concentration indices are given in Table II. Note that the closer the curve is to the diagonal, the more balanced is the distribution (small concentration). Firstly, it is interesting to observe the behavior of the baseline algorithms. The curve of *SR* shows a very big concentration bias towards the popular items, with the typical $20 : 80$ inequality that only about 20% of the most popular products in the user profile get nearly 80% of the total hits in the system. This illustrates how standard performance measures can hide important detail about the quality of the recommendation. The standard methodology for evaluating performance only reports the total precision, but the recommendation quality falls off quickly once the popularity of items decreases.

IIP shows a big advantage over the *SR* method. *UUP(b=8)* achieves a similar performance to *IIP*. After applying dimension reduction, both *SR(SVD)* and *IIP(SVD)* are biased to retrieve unpopular items, where *SR(SVD)* achieves a better total precision. Among the methods in this figure, *IIP* has the best precision with a small drop in comparison to the *SR* algorithm. However, from Table II it may be observed that *IIP(SVD)* has a big drop in total precision (comparing to *SR*) and thus the reduction in popularity bias has come at a severe cost in overall performance. In terms of precision the *UUP(b=8)* strategy does not show a competitive result.

Figure 7 depicts the concentration curves of *L1* and *KLD* algorithms proposed in Section VI, along with the baseline algorithms. For the *PR* algorithm, as expected its concentration is more extreme than the *SR* algorithm as it always recommends the most popular items. However, such a simple algorithm gives a quite good total system precision, as depicted in Table II, only a small drop from *SR* and indeed the concentration of the *SR* algorithm is not much smaller. It indicates the extent to which the performance of *SR* is achieved through recommending popular items. Given this evidence, it is not surprising when we reach out for the niche items, in general there will be a certain drop in total system precision. The *RR* algorithm presents a nearly perfect concentration curve, i.e. very close to the diagonal, but a very low precision. The *L1* and *KLD* methods produce two curves between *RR* and *SR* algorithms. It's noteworthy that no drop in precision is observed for *L1*. The *KLD* method shows an advantage over the *L1* method in terms of alleviating concentration and retrieving niche items, at the cost of a small drop in precision.

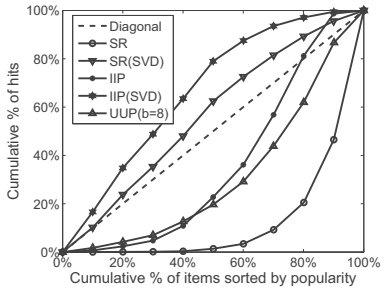

Figure 6. Concentration curves for popularity discount strategies.

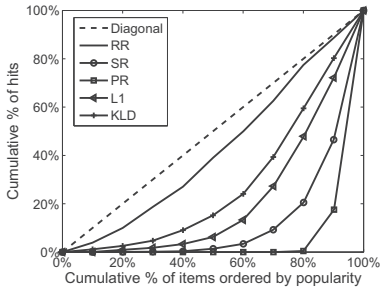

Figure 7. Concentration curves of baseline and L1, KLD algorithms.

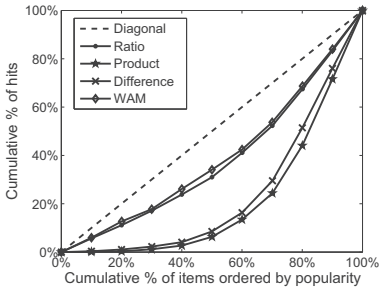

Figure 8. Concentration curves of algorithms applying other scores.

Table II
CONCENTRATION INDEX CI, GINI INDEX GI AND TOTAL PRECISION P.

	SR	SR(SVD)	PR	IIP	UUP(8)	IIP(SVD)	L1	KLD	Ratio	Product	Difference	WAM
CI	0.75	-0.14	0.87	0.3	0.3	-0.35	0.57	0.43	0.25	0.58	0.52	0.21
GI	0.84	0.62	0.87	0.8	0.83	0.71	0.72	0.69	0.7	0.71	0.68	0.69
P	0.12	0.073	0.093	0.086	0.044	0.051	0.114	0.107	0.067	0.129	0.126	0.071

Looking at Figure 8, the *Product* and *Difference* methods give similar results to the *L1* method, i.e. only a modest improvement in CI. Nevertheless, they even produce a certain improvement in the overall system precision over SR (14.5% improvement for *Product* and 13.3% for *Difference*). The *Ratio* and *Ratio* methods have a quite different behaviour. For them a remarkable improvement in CI is obtained with a bigger cost of precision (however, still a great advantage over *RR*). The *Ratio* method outperforms the *Ratio* method in both metrics, i.e. better precision and less concentration.

The Gini index (GI) for each algorithm is also presented in Table II. *SR(SVD)* provides the best performance here with a 25.5% improvement over *SR*, while the *Difference* algorithm achieves 18% improvement. Although the improvement seems modest in comparison to CI, it still can be regarded as a big improvement since this indicates a more even hits across all items in the entire catalogue. To sum up, *Ratio* is the best in removing the popularity bias, while *SR(SVD)* is the best in removing the overall concentrations.

In terms of system efficiency, the computational complexity for the *IIP* strategy is linear to the total number of items, $O(n)$. For the *UUP* strategy, as reported in [14] it takes $O(nm)$ to obtain the profile utility for each user over all the items they rated. For the strategy applying probabilistic scores, with s prepared the extra step for training $p_L(s)$ is to go over the users and see for which ranges of s the relevant items fall in, which takes $O(m)$. It can be computed off-line and is easily updated when new items are rated or new users are introduced. No exponential or high power of computational complexity is observed.

B. Discussion

Table III provides an example of two top-10 recommendation lists generated by the *SR* and *Ratio* methods respectively, for a randomly selected user. The popularity of the items is presented. Firstly, the problem of concentration among popular items is well observed for the list generated by the *SR* algorithm. Very popular movies such as the *Star Wars* series generate a huge amount of hits (i.e. relevant recommendations) for the system but have a very little *value*, since they are already well known. All the items in the list recommended by the *SR* method are such kind of movies (with popularity bigger than $1,000$). If it is accepted that the role of a recommender system should be more in discovering relevant items that the user is not aware of, rather than advertising well-known items over and over again, the *Ratio* method can be seen to out-perform *SR*. Niche and relevant items have been brought in to the recommendation. It is very possible that the user is not aware of them before because they are not generally popular. Nevertheless, their relevance is assured by the maintenance of the total system precision. The second list actually provides a mix of popular and niche items, which can be regarded as a reasonable strategy, in the sense of providing the customers some familiar items as well as some surprises. The duplicates of the very popular movies are removed in the new list, which is also a cause of the precision loss. For example, only *Episode V* of the *Star Wars* is kept and being aware of its presence in the system, the user can easily search for the other episodes.

Moreover, a side-effect we found is, bringing in the niche items actually increases the *set diversity* of the recommendation list. It is defined as the average of pairwise dissimilarity of all the items in the recommendation set [17], [18]. Some work such as [19] has followed up to maximise the set diversity. The diversity of the recommendation set has improved by 47% (0.48 for *SR* and 0.71 for *Ratio*), averaging over all users. Thus, while the approach of this paper has focused on extending the recommendation to less popular

Table III
COMPARISON OF THE RECOMMENDATION LISTS FROM SR AND RATIO.

List1 (SR)	Popularity	List2 (Ratio)	Popularity
Star Wars: Episode V...(1980)	1516	Rocky II (1979)	247
Star Wars: Episode IV...(1977)	1518	Philadelphia (1993)	305
Pulp Fiction (1994)	1088	The Sixth Sense (1999)	1240
Raiders of the Lost Ark (1981)	1295	Pulp Fiction (1994)	1088
The Matrix (1999)	1322	Short Cuts (1993)	152
The Shawshank Redemption(1994)	1091	Blood Simple (1984)	315
The Terminator (1984)	1092	Star Wars: Episode IV...(1977)	1518
Back to the Future (1985)	1321	Blade Runner (1982)	920
Terminator 2: Judgment Day (1991)	1368	Mulholland Falls (1996)	142
Star Wars: Episode VI...(1983)	1446	The Usual Suspects (1995)	897

items, an effect is that the items in the recommendation set tend to be less alike.

VIII. CONCLUSIONS

In this work we investigate the problem of retrieving niche items with an aim of improving end-users' satisfaction. Initially, an analysis is performed on real consumption data to present users' preference curves and demonstrate the importance of niche items to a majority of users, rather than just a few users. We emphasise that to provide a good quality niche recommendation it is not enough to provide a range of different recommendations. Rather, we must ensure that these recommendations are relevant to the end user. Thus, we measure any improvements in recommendation of niche products in conjunction with the overall number of successful recommendations, as given by the system precision. In summary, an evaluation on the Movielens dataset demonstrates that the methods proposed in this paper succeed in retrieving niche and relevant items. Some algorithms with a modest improvement in concentration even obtain a better precision as well, while some others that achieve a great improvement in concentration have a certain loss in precision.

REFERENCES

[1] C. Anderson. The long tail. *Wired Magazine*, 12(10):170–177, 2004.

[2] C. Anderson. The long tail: Why the future of business is selling less of more. *Hyperion*, 2006.

[3] M. Berry, S. Dumais, and T. Letsche. Computational methods for intelligent information access. In *Proceedings of Supercomputing'95*, 1995.

[4] E. Brynjolfsson and Y. J. Hu. Goodbye pareto principle, hello long tail: The effect of search costs on the concentration of product sales. *MIT Center for Digital Business Working Paper*, 2007.

[5] R. J. Budzynski, W. Kondracki, and A. Krolak. Applications of distance between probability distributions to gravitational wave data analysis. *Classical and Quantum Gravity*, 25, 2008.

[6] O. Celma and P. Herrera. A new approach to evaluating novel recommendations. In *Proceedings of RecSys'08*, pages 179–186, 2008.

[7] T. M. Cover and J. A. Thomas. Elements of information theory. *John Wiley*, 1991.

[8] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems*, 22(1):143–177, 2004.

[9] A. Elberse. Should you invest in the long tail? *Harvard Business Review*, 86(7/8):88–96, 2008.

[10] A. Elberse and F. Oberholzer-Gee. Superstars and underdogs: An examination of the long tail phenomenon in video sales. *Harvard Business School Working Paper*, 7(15), 2006.

[11] D. Fleder and K. Hosanagar. Recommender systems and their impact on sales diversity. In *Proceedings of the 8th ACM conference on Electronic commerce*, pages 192–199, 2007.

[12] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD Conference*, 2008.

[13] J. L.Herlocker, J. A.Konstan, L. G.Terveen, and J. T.Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, 2004.

[14] M.O'Mahony, N. Hurley, and G.Silvestre. Utility-based neighbourhood formation for efficient and robust collaborative filtering. In *Proceedings of ACM Conf. on Electronic Commerce*, pages 17–20, 2004.

[15] C. S. Moskowitz, V. E. Seshan, and E. R. Riedel. Estimating the empirical lorenz curve and gini coefficient in the presence of error with nested data. *Statistics in Medicine*, 27:3191–3208, 2008.

[16] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Application of dimensionality reduction in recommender systems. In *ACM WebKDD Workshop*, 2000.

[17] B. Smyth and P. McClave. Similarity vs. diversity. In *Proceedings of the 4th International Conference on Case-Based Reasoning*, pages 347–361, 2001.

[18] M. Zhang and N. Hurley. Avoiding monotony: Improving the diversity of recommendation lists. In *Proceedings of 2nd ACM Recsys*, 2008.

[19] M. Zhang and N. Hurley. Novel item recommendation by user profile partitioning. In *Proceedings of the 2009 IEEE/WIC/ACM International Conferences on Web Intelligence*, pages 508–515, 2009.