Battling Predictability and Overconcentration in Recommender Systems

Sihem Amer-Yahia[†], Laks V.S. Lakshmanan[‡], Sergei Vassilvitskii[†], Cong Yu[†]
[†] Yahoo! Research, New York, NY, USA
{sihem,sergei,congyu}@yahoo-inc.com
[‡] University of British Columbia, Vancouver, BC, Canada
laks@cs.ubc.ca

1 Introduction

Today's recommendation systems have evolved far beyond the initial approaches from more than a decade ago, and have seen a great deal of commercial success (c.f., Amazon and Netflix). However, they are still far from perfect, and face tremendous challenges in increasing the overall utility of the recommendations. These challenges are present in all stages of the recommendation process. They begin with the *cold start* problem: given a new user how do we recommend items to the user without forcing her to give feedback on a starter set of items. Similarly, given a new item introduced into the system, how do we know when (and to whom) we should recommend it.

A different problem is that of *data sparsity*. Although every system has 'super' raters, that is people who rate thousands (and in some cases tens of thousands) of items, the number of such people is low. In fact, most people give feedback on only a handful of items, resulting in a large, but very sparse dataset. The data also exhibits the long tail phenomenon, with the majority of items getting just a handful of ratings. However, as recent studies have shown, understanding the tail is crucial—while no item in the tail is rated by too many people, the vast majority of people rate at least a few items in the tail [8].

In this work, we focus on a different question facing recommender systems: what determines the utility of a recommendation? In the past, accuracy, usually measured on some held out dataset, has served as the gold standard; indeed this was the only measure used by the famed Netflix prize. However, we must be careful to remember that accuracy is not identical to relevancy or usefulness of a recommendation. Accuracy is the leading component—if the accuracy of recommendations is low the system is not trusted by the user. However, once a recommender system achieves some accuracy bar, obtaining even higher accuracy does not necessarily make a recommendation more relevant. Indeed, recent work (e.g., see Sharma and Carenini [12]) has criticized the use of mean absolute error (MAE) as the sole measure of effectiveness of a recommender system and has proposed alternative measures inspired by decision theory. In this paper, we take a different tack and contend that once an accuracy bar (which may be based on a measure such as MAE) is achieved, *diversity* and *freshness* of recommendations play a crucial role in making a recommendation more relevant and useful to a user.

Diversity One of the problems plaguing recommender systems overly focused on accuracy is *overconcentration*. This is the problem of recommending a set of items, in which all of the items are too similar to each other. For example, the system may learn that a user Alice has a certain penchant for fantasy books and may

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

recommend other books from this specific genre. While Alice surely likes most of these books, she may also enjoy reading detective stories. However, if her tastes in detective stories are more finicky, an accuracy-driven system would never take the risk in recommending an Agatha Christie book that Alice may or may not like over a Lewis Carrol novel that she is sure to enjoy. The result of never taking such risks is a series of recommendations that are very homogeneous. As noted in [3], a series of recommendations that look too similar to each other could easily turn off the user and lower her interest in the site over time.

In this case, the solution is to diversify the set of results, that is to insist that the items returned by the recommendation system are sufficiently different from one another. How do we ensure that the recommended items are different from each other? If intrinsic properties of items, in the form of attributes, are available, we can check that they differ from each other on some attributes. If these are not available or expensive to compute (e.g., images, video, etc.), we must turn to other means. This is a topic we address in this paper and argue that *explanations* of recommendations constitute an excellent piece of information with which to measure item (dis)similarity. It is worth noting that augmenting recommendations with explanations has been recognized as an important step in building user trust and adoption [3] and generating explanations has been recognized as one of the outstanding open problems for recommender systems [9].

Freshness A more subtle problem, not usually addressed is that of *predictability*. Even if the set of results returned by the system is diverse, i.e., the items being recommended are quite different from one another, it may still lack *serendipity*, that is, be already known to the user. Imagine, for example, a system that only recommends best sellers from every genre. While such a system would likely have high accuracy scores (these items are best sellers for a reason), and the set of recommended items is diverse (since recommendations come from many different genres), such a system would rarely be useful. We must be cognizant of the fact that the recommender system is not the sole way by which users learn of new items. In fact, they watch television, they ask their friends for advice, etc. It is generally believed that there is no need to recommend best sellers, precisely because a typical user already knows about these books, however many systems simply draw the line at best sellers without trying to estimate the likelihood that the user has already been exposed to an item through other means.

In order to better estimate this probability, we begin by dividing the item space into a set of *regions*. We think of regions as representing different broad types of items (for example genres of movies), however they need not have any semantic correlation, and can be produced automatically by a separate classifier. We then define the *stickiness* of a user to a region, which describes how often a user rates an item from that region, as compared to a background distribution. Intuitively, if a user (or the user's network) has high stickiness for a region, she is likely to be familiar with the items in that region (since she is likely to have been exposed to them through her network), and hence, is not in need for further recommendations from that region. On the other hand, a user having below average stickiness to a region can be a signal of one of two actions: either the user *likes* these items but is not familiar with them, in which case the system should recommend an item from this region, or the user knows these items, but does not like them. To distinguish between the two cases we rely on the fact that we started with a high accuracy system, and recommend only items with relevance scores above a certain threshold.

Diversity vs. Freshness Notice that the problems related to (lack of) diversity and (lack of) freshness, although closely related, are quite independent. You can have a system making recommendations that are sufficiently dissimilar from each other and yet are predictable for a given user and vice versa. As pointed out above, recommendations consisting of bestsellers, while being diverse, may be utterly predictable for most users. On the other hand, a series of mystery recommendations to Alice in our example above, while fresh in that she may not be sufficiently exposed to mystery books, may end up in recommendations too similar to each other. Unfortunately, in the recommender systems literature, these two problems are conflated and are often referred to using a single term (lack of) diversity. Indeed, one of the approaches suggested for solving this problem is randomization [3]. More precisely, injecting a few random choices into the set of recommendations generated by a system is believed to make the recommendations diverse. With respect to the issues of diversity and freshness (as defined in this work), randomization may yield results that have higher diversity and higher

freshness, although *there is no guarantee.* In this paper, we make a clear distinction between these two problems and summarize some of our recent work that addresses both of them.

2 Recommendation Strategies Overview

In this section, we review the two most basic recommendation approaches: *item-based* and *collaborative filtering* [3]. Despite having been around for a long time, these two approaches are still at the core of the most recent advancements in recommendation systems.

Item Based Strategies Earlier recommendation strategies are mostly item based. They aim to recommend items similar to the ones the end-user preferred in the past. The rating of an item $i \in \mathcal{I}$ (the set of all items) by a current user $u \in \mathcal{U}$ (the set of all users) is estimated as follows: $\texttt{relevance}(u, i) = \Sigma_{i' \in \mathcal{I}} \texttt{ItemSim}(i, i') \times \texttt{rating}(u, i')$.

Here, $\texttt{ItemSim}(i, i')$ returns a measure of similarity between two items i and i', and $\texttt{rating}(u, i')$ indicates the rating of item i' by user u (it is 0 if u has not rated i'). Item based strategies are very effective when the given user has a long history of tagging/rating activities. However, it does have two drawbacks. First, items recommended by item based strategies are often very similar to what the user already knows [10] and therefore other interesting items that the user might like have little chance of being recommended. Second, item based strategy does not work well when a user first joins the system. To partially address those drawbacks, collaborative filtering strategies have been proposed.

Collaborative Filtering Strategies Collaborative filtering (CF) strategies aim to recommend to an end-user items which are highly rated by users who share similar interests with or have declared relationships with her. CF strategies can be classified into two categories: memory-based and model-based. Memory-based CF startegies are similar in spirit to, and can be regarded as dual of, item-based strategies. Here, the rating of an item i by the end-user u is estimated as follows: $\texttt{relevance}(u, i) = \Sigma_{u' \in \mathcal{U}} \texttt{UserSim}(u, u') \times \texttt{rating}(u', i)$.

Here, $\texttt{UserSim}(u, u')$ returns a measure of similarity or connectivity between two users u and u' (it is 0 if u and u' are not connected). Collaborative filtering strategies broaden the scope of items being recommended to the user and have become increasingly popular. Note that in both strategies, we use $relevance(u, i)$ to denote the estimated rating of i by u.

In model-based CF strategies, a model is built, using a machine learning technique such as Bayes Network, clustering, or using association rules, and the model is used for making predictions of the rating of an item by a given user. See [3, 9] for an excellent survey of various recommendation algorithms.

3 Battling Overconcentration through Diversification

The problem of overconcentration can be addressed by *diversification*, where the goal is to ensure that the set of recommended items are dissimilar with each other, but nonetheless relevant to the user's interests. Recommendation diversification is well studied in the recommendation systems community. However, to the best of our knowledge, almost all of the techniques proposed so far for recommendation diversification [10] are *attribute-based*. In other words, the diversity of the recommended list is defined by how much each item in the list differs from the others in terms of their attribute values. For example, in Yahoo! Movies, recommended movies are post-processed based on the set of attributes that are intrinsic to each movie (such as genre, director, etc.). Typically, a heuristic distance formula is defined on the set of attributes in order to allow the system to quantify the distance between a pair of items. In [15], diversity is measured based on a specific attribute of the item which is defined by the taxonomy in which the item is classified. This can also be considered as attribute-based since the classification of each item is intrinsic to that item.

3.1 Drawbacks of Using Attribute-Based Diversification

While suitable for traditional recommender systems where the items are well-described by their attributes (e.g., books in Amazon and movies in Netflix), attribute-based diversification has two main drawbacks.

First, there can be a *lack of attribute descriptions for items*. Particularly, items in social network sites often *lack a comprehensive set of attributes*, unlike their counterparts in traditional recommender systems. For example, URLs in del.icio.us are only described by the set of tags associated with them, so are images in Flickr and videos in YouTube. Tags are attached to the items by individual users and are fundamentally different from intrinsic descriptions of the items. Analyzing intrinsic properties of these items, on the other hand, is extremely difficult or computationally prohibitive. For example, to obtain intrinsic descriptions of each URL in del.icio.us, one is required to crawl and analyze the web page. This is very costly to do for the mere purpose of diversification and certainly beyond the purpose and capability of del.icio.us. For multimedia sites like Flickr and YouTube, while image processing and video analysis techniques do exist, the effort required to extract feature vectors for images and videos may be considered too high a price to pay just for diversifying recommendations.

Second, there can be *substantial computational overhead due to attribute retrieval for the purpose of diversification*. A diversification approach based on attributes often requires accessing item attributes in order to perform the diversification: a step that can become costly when the database is very large as in many of the collaborative tagging sites, especially when the attributes are not leveraged by the recommendation strategies (e.g., many collaborative filtering strategies).

3.2 Explanation-Based Diversification

To overcome the above drawbacks, we describe the novel approach of explanation-based diversification introduced in [14, 13]. To begin with, we denote by $\texttt{RecItems}(u)$, the set of candidate recommended items generated by one of the recommendation strategies described above. The size of this set is typically larger than the final desired number of recommendations.

An explanation for a recommended item depends on the underlying recommendation strategy used. If an item i is recommended to user u by a content-based strategy, then an *explanation* for recommendation i is defined as: $\texttt{Expl}(u,i) = \{i' \in \mathcal{I} \mid \texttt{ItemSim}(i,i') > 0 \ \& \ i' \in \texttt{Items}(u)\}$, i.e., the set of items similar to items (i') that user u has rated in the past. The explanation may contain more information such as the similarity weight $\texttt{ItemSim}(i,i') \times \texttt{rating}(u,i')$. If an item i is recommended to user u by a collaborative filtering strategy, then an *explanation* for a recommendation i is: $\texttt{Expl}(u,i) = \{u' \in \mathcal{U} \mid \texttt{UserSim}(u,u') > 0 \ \& \ i \in \texttt{Items}(u')\}$, i.e., the set of users similar to u who have rated item i. Similarly, we can augment each user u' with the similarity weight $\texttt{UserSim}(u,u') \times \texttt{rating}(u',i)$.

Note that in all cases, the explanation of a recommendation is either a set of items or a set of users. Based on this, we can define diversity of recommendations as follows. Let i, i' be a pair of items (recommended to user u) with associated explanation sets $\texttt{Expl}(u,i)$ and $\texttt{Expl}(u,i')$. Then the *diversity distance* between i, i' for user u can be defined as a similarity measure based on standard metrics such as Jaccard similarity coefficient or cosine similarity. E.g., the *Jaccard diversity distance* between recommendations i and i' is:

$$DD_u^J(i,i') = 1 - \frac{|\texttt{Expl}(u,i) \cap \texttt{Expl}(u,i')|}{|\texttt{Expl}(u,i) \cup \texttt{Expl}(u,i')|}.$$

Note that this is defined as the complement of the standard Jaccard coefficient, since we want to use this as a distance measure. Intuitively, in the case of collaborative filtering, the distance between items depends on the ratio between the number of users who recommend both items and the total number of users who recommend these items.

When weights are incorporated, the *cosine diversity distance* between recommendations i and i' is defined by treating the explanations $\texttt{Expl}(u,i)$ and $\texttt{Expl}(u,i')$ as vectors in a multi-dimensional space and defining $DD_u^C(i,i')$ as 1 minus standard cosine similarity between the vectors. We refer the readers to the Vector Space Model [1] for more details. In the case of collaborative filtering, the weighted diversity distance depends on

the ratio between the number of similar users who recommend both items and the total number of users who recommend these items. When we want to be neutral with respect to the type of diversity distance measure used and opt for the notation $DD_u(i, i')$. Depending on the context, it represents $DD_u^J(i, i')$ or $DD_u^C(i, i')$.

For a set of items $S \subseteq \texttt{RecItems}(u)$, we define: $DD_u(S) = avg\{DD_u(i, i') \mid i, i' \in S\}$, i.e., $DD_u(S)$ is the average diversity distance between pairs of items in the set S. Algorithms can then be designed to leverage this distance measure to strike a good balance between relevance and diversity in the recommendations. In our previous work [13], we describe two efficient heuristic algorithms for achieving this balance: *Algorithm Swap* and *Algorithm Greedy*.

3.3 Brief Description of Diversification Algorithms

The basic idea behind **Algorithm Swap** is to start with the K highest scoring items, and swap the item which contributes the least to the diversity of the entire set with the next highest scoring item among the remaining items. At each iteration, a candidate item with a lower relevance is swapped into the top-k set if and only if it increases the overall diversity of the resulting set. To prevent a sudden drop of the overall relevance of the resulting set, an optional pre-defined upper-bound `UB` (on how much drop in relevance is tolerated) can be used to stop the swapping when the lowest relevance of the remaining items is no longer high enough to justify the swap. The selection of the item to be swapped is accomplished by searching over all items in the current top-K set S and picking item i with the minimum diversity. More precisely, let $D_S^i = \sum_{j \in S, j \neq j} DD_u(i, j)$, we pick the item $i \in S$ with the minimum D_S^i as the candidate for swap.

Another heuristic approach is to start with the most relevant item, greedily add the next most relevant item if and only if that item is far enough away (compared with a distance bound) from existing items in the set, and stop when there are K items. The drawback, however, is that the distance bound is often hard to obtain and can vary from user to user. To address this problem, **Algorithm Greedy** relies on iteratively refining two diversity thresholds: an upper-bound `UB`, initially set to 1, and a lower-bound, `LB`, initially set to 0. The algorithm first iterates through the set of candidate items in the order of their relevances and generates two lists: `DivList`, and `SimList`. The `DivList` contains items that are all maximally distant from each other, while the `SimList` contains items that have zero distance to some items in `DivList`. Because items are accessed in the order of their relevance, `SimiList` essentially contains items that we no longer consider. If `DivList` already contains enough items, we pick K most relevant items from it can return. If not, we adjust `UB` and `LB` and reiterate through the candidate items. At each iteration, the `KeepList` tracks items that will definitely be in the result set while the `DiscardList` tracks items that should be eliminated from consideration. These two lists are merged with `DivList` and `SimList`, respectively, at the end of each iteration. The algorithm stops when K are found in `DivList` or the difference between `UB` and `LB` drops below a threshold.

4 Battling Predictability through Thinking Outside-The-Box

As stated in the introduction, most previous work focuses on increasing diversity to address predictability (see [11] and [15].) For example, DailyLearner [7], a content-based recommender system, filters out highly relevant items which are too similar to items the user has rated in the past. In our work [2], we develop a formal solution to recommendation predictability which relies on identifying *item regions* that guide the system in *taking some risk* to help users make *fresh discoveries*, while *maintaining high relevance*. We refer to our solution as $\mathcal{OTB}$, for Outside-The-Box.

In [2], we experimentally show two important benefits of our approach. First, $\mathcal{OTB}$ recommendations are of high quality as measured by standard metrics. Second, they differ significantly from traditional ones, and this difference is *not* simply in the tail end of the recommendations—we find that they contribute significantly to overall quality. This argues that $\mathcal{OTB}$ recommendations are complementary to conventional ones, and that both

novelty and relevance are necessary to achieve user satisfaction.

4.1 Going Outside the Box

A *region* (i.e., the "box") is a group of similar items. $R^{\mathcal{I}}$ denotes a (potentially overlapping) assignment of items $\mathcal{I}$ into regions. Regions in $R^{\mathcal{I}}$ are produced using similarity distances between items. We explore two such similarities: *attribute-based* and *activity-based*. Let A be a set of item attributes, also called dimensions. Attribute-based similarity relies on a function d_A. For any two items i and j the distance $d_A(i,j) = 0$ iff $\forall a \in A$ we have $i.a. = j.a$, and $d_A(i,j) = \infty$ otherwise.

For movies, region dimensions may include movie attributes like genre and directors. A region instance is often identified by its dimensions and their values (e.g., {(genre=comedy), (producer=Disney)}).

Activity-based similarity produces regions which identify items rated by a large enough number of similar users. More formally, for any two items i and j, and action a let $a(i)$ and $a(j)$ define the respective sets of users that performed action a on the item. Then let $d(i,j)$ be the Jaccard dissimilarity between $a(i)$ and $a(j)$:

$$d(i,j) = 1 - \frac{|a(i) \cap a(j)|}{|a(i) \cup a(j)|}$$

To produce an assignment of items into regions given a similarity function, we turn to the k-means^{++} algorithm [6]. The algorithm is known to converge quickly, and for clustering n items requires only $O(nkr)$ distance computations, where k is the number of clusters and r is the number of rounds performed by k-means.

4.1.1 Region $\mathcal{OTB}$-ness

We let $\texttt{items}(u)$ denote the set of items that are rated by u, $\texttt{items}(r)$ the set of items belong to region r, $\texttt{items}(u,r)$ the set of items belong to region r that are rated by u, i.e., $\texttt{items}(u,r) = \texttt{items}(u) \cap \texttt{items}(r)$ and, $\texttt{rating}(u,i)$ the known rating of user u for item i;

The *stickiness of a user u to a region r*, $\texttt{stick}(u,r)$ is the fraction between the number of items rated by u which belong to r over the total number of items rated by u. That is, $\texttt{stick}(u,r) = \frac{|\texttt{items}(u,r)|}{|\texttt{items}(u)|}$. For example, a user who rated 500 movies, 50 of which are Drama, would have a stickiness of 10% for the region {(genre=Drama)}. Intuitively, stickiness measures the degree of familiarity of a user toward a given region: the higher the stickiness, the more likely the user already knows about items within the region. Note that if the given region is the entire set of items ($\mathcal{I}$), then $\texttt{stick}(u,\mathcal{I}) = 1$ for any user u.

Similarly, we define the stickiness of a group of users (i.e., a network) N to a region r, $\texttt{stick}(N,r)$ is the average of each individual member's stickiness. Hence, $\texttt{stick}(N,r) = \frac{1}{|N|}\Sigma_{u \in N}(\texttt{stick}(u,r))$. Furthermore, we have the deviation of stickiness:

$$\texttt{stickDev}(N,r) = \sqrt{\frac{1}{|N|}\Sigma_{u \in N}(\texttt{stick}(u,r) - \texttt{stick}(N,r))^2}.$$

The network stickiness measures the familiarity toward the given region by a group of users collectively. The deviation of stickiness measures how consistent each member's stickiness is with the others. The lower the deviation, the more likely every member in the group is familiar (or unfamiliar) with items in the given region. When N is the entire group of users ($\mathcal{U}$), we have the global stickiness, $\texttt{stick}(\mathcal{U},r)$, and deviation, $\texttt{stickDev}(\mathcal{U},r)$, for the region.

There are two main factors in measuring a region's $\mathcal{OTB}$-ness for a given user: the level of unfamiliarity and the (under-)exposure potential. We combine those two factor in the definition of $\mathcal{OTB}$-ness and define the $\mathcal{OTB}$-ness for a region r by a given user u as $\texttt{otb}(u,r) = \texttt{otbBase}(u,r) \times \texttt{otbFactor}(u,r)$. where the *level of unfamiliarity* for r by u is defined as:

$$\texttt{otbBase}(u,r) = \frac{\texttt{stick}(\mathcal{U},r) - \texttt{stick}(u,r)}{\texttt{stickDev}(\mathcal{U},r)},\ \text{if } \texttt{stick}(\mathcal{U},r) > \texttt{stick}(u,r),$$

and 0 otherwise. And the *exposure factor* for r by u is defined as:

$$\texttt{otbFactor}(u,r) = \frac{\texttt{stick}(\mathcal{U},r) - \texttt{stick}(N,r)}{\texttt{stickDev}(\mathcal{U},r) + \texttt{stickDev}(N,r)} \times 2,\ \text{if } \texttt{stick}(\mathcal{U},r) > \texttt{stick}(N,r),$$

and 0 otherwise, where N is u's network. Here, normalization by the global deviation in `otbBase` is done to identify the regions whose unfamiliarity are the most statistically significant. And, a region has a high `otbFactor` if the user's network is unfamiliar with items in the region.

4.1.2 Region-Based Relevance

Identifying good items within $\mathcal{OTB}$ regions now becomes a challenge since neither the user nor the user's network knows much about items in those regions with a high $\mathcal{OTB}$-ness for that user (according to the definitions in Section 4.1.1). As a result, computing the user's expected rating, i.e., relevance, for items within those regions requires special attention. To address this question, we propose the notion of *region-region correlation* to identify the set of regions that *implies* each $\mathcal{OTB}$ region. We then construct an *expanded region network*, which consists of users who are similar to the target user based on items in those correlated regions.

We use association rules [5] to identify region-region correlations of the form $r \Rightarrow r'$ where r and r' are different regions in $R^{\mathcal{I}}$. A region s is a source region of a region r if and only if at least $x\%$ of the users who rate items in s also rate items in r, where x is a custom defined threshold. Source regions indicate general trends such as *people who rate Woody Allen movies also rate David Lynch movies*. We use $\texttt{sources}(r)$, to denote the set of all source regions of a region r. Based on this, $\texttt{exSim}(u,u',r)$, the expanded similarity of two users given a region can be defined as:

$\texttt{exSim}(u,u',r) = max_{r' \in \texttt{sources}(r)} \texttt{userSim}(u,u',r')$

where $\texttt{userSim}(u,u',r)$ is a similarity between two users restricted to region r, formally defined as:

$$\texttt{userSim}(u,u',r) = \frac{|\{i | i \in \texttt{items}(u,r) \ \wedge\ i \in \texttt{items}(u',r) \ \wedge\ |\texttt{rating}(u,i) - \texttt{rating}(u',i)| \leq 2\}|}{|\{i | i \in \texttt{items}(u,r) \ \vee\ i \in \texttt{items}(u',r)\}|}$$

where two ratings with a difference less than 2 (on a scale of 0 to 5) is considered to be similar. This is number is user customizable and chosen based on our experience.

The expanded region network, $\texttt{exNet}(u,r)$, for a user u and a region r is the set of users $u' \in \mathcal{U}$ such that $\texttt{exSim}(u,u',r) \geq \theta$ where θ is an application-dependent threshold. Intuitively, $\texttt{exNet}(u,r)$ is the set formed by users who share similar interests with u over source regions of r. We can now have:

$\texttt{relevance}(u,r,i) = \Sigma_{u' \in \texttt{exNet}(u,r)} \texttt{exSim}(u,u',r) \times \texttt{rating}(u',i)$.

4.2 Consolidation

Finally, we define the *overall score* of an item i as follows:

$$\texttt{overall}(u,i) := \Sigma_{r \in \texttt{regions}(i)} \texttt{otb}(u,r) \times \texttt{relevance}(u,r,i)$$

where $\texttt{regions}(i)$ is the set of all regions an item belongs to, $\texttt{otb}(u,r)$ denotes the $\mathcal{OTB}$ score of region r for user u and $\texttt{relevance}(u,r,i)$ is the region-specific relevance score of item i for user u.

5 Summary and Open Problems

The first hurdle facing any recommender system is that of accuracy, a system that cannot predict a user's affinity for an item cannot provide useful recommendations. However, we argue that once an acceptable accuracy bar (based on standard measures such as MAE) is reached, the system must begin to pay attention to other metrics such as diversity and freshness. Otherwise a system maximizing accuracy leads to recommendations that are predictable and boring, which adversely affect user loyalty.

Diversification is a clear answer; however, in the recommender systems literature, the term diversity conflates solutions to two related, but very different, problems. We distinguish between these and refer to them as overconcentration and predictability. In the former, the system recommends a homogeneous set of items. We tackle this problem by generating recommendations that are dissimilar to each other by leveraging explanations associated with recommendations. The latter has the system being overly safe, and recommending items that the user is likely to already be familiar with. Our approach for ensuring freshness is based on modeling what regions of the item space a user, her network, and the entire network, has been exposed to, and then generating recommendations that take this information into account.

The algorithms and approaches we described in this paper are applicable to memory-based recommender systems. Extending these approaches for model-based algorithms is an important open problem. More recently, there has been a push for context-aware recommender systems [4]. Understanding the interplay between the issues of diversity and freshness and the notion of context in such recommender systems is another interesting direction for future work.

References

[1] `http://en.wikipedia.org/wiki/Vector_space_model`

[2] Z. Abbassi, S. Amer-Yahia, L. V. S. Lakshmanan, S. Vassilvitskii, C. Yu, *Getting recommender systems to think outside the box*, in Proc. of the Recommender Systems Conference (RecSys), 285–288, 2009.

[3] G. Adomavicius, A. Tuzhilin, *Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions*, in IEEE Trans. Knowl. Data Eng., (17):6, 2005.

[4] G. Adomavicius, A. Tuzhilin, *Context-aware recommender systems*, in ACM Int. Conf. Recommender Systems, 2008.

[5] R. Agrawal, R. Srikant, *Fast Algorithms for Mining Association Rules in Large Databases*, in VLDB, 1994.

[6] D. Arthur, S. Vassilvitskii, *K-Means++: the advantages of careful seeding*, in SODA, 2007.

[7] D. Billsus, M. Pazzani, *User Modeling for Adaptive News Access*, in User Modeling and User-Adapted Interaction, (10):2/3, 147–180, 2000.

[8] S. Goel, A. Broder, E. Gabrilovich, B. Pang, *Anatomy of the Long Tail: Ordinary People with Extraordinary Tastes*, in WSDM, 2010.

[9] Y. Koren, *Recent Progress in Collaborative Filtering*, in ACM Int. Conf. Recommender Systems, 2008.

[10] S. M. McNee, J. Riedl, J. A. Konstan, *Being accurate is not enough: how accuracy metrics have hurt recommender systems*, in CHI Extended Abstracts, 2006.

[11] S. A. Munson, D. X. Zhou, P. Resnick, *Sidelines: An Algorithm for Increasing Diversity in News and Opinion Aggregators*, in AAAI, 2009.

[12] R. Sharma, G. Carenini, *Exploring more Realistic Measures for Collaborative Filtering*, in AAAI, 2004.

[13] C. Yu, L. Lakshmanan, S. Amer-Yahia, *It Takes Variety to Make a World: Diversification in Recommender Systems*, in Int'l Conf. on Extending Database Technology (EDBT), 2009.

[14] C. Yu, L. Lakshmanan, S. Amer-Yahia, *Recommendation Diversification Using Explanations*, in ICDE, 2009.

[15] C. N. Ziegler, S. M. McNee, J. A. Konstan, G. Lausen, *Improving Recommendation Lists Through Topic Diversification*, in Proc. of the World Wide Web Conference (WWW), 2005.