Recommendations Meet Web Browsing: Enhancing Collaborative Filtering using Internet Browsing Logs

Royi Ronen
Microsoft Israel
royir@microsoft.com

Elad Yom-Tov
Microsoft Research Israel
eladyt@microsoft.com

Gal Lavee
Microsoft Israel
galla@microsoft.com

***Abstract*—Collaborative filtering (CF) recommendation systems are one of the most popular and successful methods for recommending products to people. CF systems work by finding similarities between different people according to their past purchases, and using these similarities to suggest possible items of interest. In this work we show that CF systems can be enhanced using Internet browsing data and search engine query logs, both of which represent a rich profile of individuals' interests.**

We introduce two approaches to enhancing user modeling using these data. We do not assume the existence of explicit ratings, but rather rely on unweighted, positive signals, that are available in most commercial contexts. We demonstrate the value of our approach on two real datasets each comprising of the activities of tens of thousands of individuals. The first dataset details the downloads of Windows Phone 8 mobile applications and the second - item views in an online retail store. Both datasets are enhanced using anonymized Internet browsing logs.

Our results show that prediction accuracy is improved by up to 72%. This improvement is largest when building a model which can predict for the entire catalog of items, not just popular ones. Finally, we discuss additional benefits of our approach, which include: improved recommendations for users with few past purchases and enabling recommendations based on short-term purchase intent.

I. Introduction

Recommendation (or recommender) systems are information filtering systems designed to suggest products or services which would be of interest to people. One of the most popular approaches to making recommendation is that of Collaborative Filtering (CF) [1]. CF-based recommendation systems work by defining a similarity between people according to their product preference. Each user is then presented with recommended items consumed by other users similar to themselves.

CF systems utilize information of the products consumed by users to make their recommendations. This can be considered a strength of the approach, as the most pertinent information for making recommendations is the consumption of the items from the catalog. However, this can also be considered a weakness, as a definition of similarity between users based solely on their consumption of a particular class of items is very limited. Furthermore, this approach makes it difficult to provide good item recommendations to users who are new to the system and have made few, if any, purchases (the so-called "cold users"). The problem of making recommendations to new users is especially troublesome in many real-world recommendation systems, where the distribution of the number of items consumed by each user is heavy tailed: most people buy few products, and only a few buy many of them. Similarly, when new items (so-called "cold items") are introduced into the item catalog, it is hard for CF recommendation systems to recommend these items. Another shortcoming of CF recommendation systems is that they have limited capacity to consider temporal changes in user interests, particularly short-term purchase intent.

Internet activity logs are the time-stamped list of sites visited by users as they browse the Internet. Internet activity logs are usually collected from users who have agreed to sharing these data through toolbars and similar browser add-ins. Search engine logs are the recordings of all queries submitted to an Internet search engine. Internet activity logs and queries to search engines have long been known to represent the physical and virtual behaviors of people [2]. As such, they have been used to track disease distributions [3], identify users with specific ailments [4] and pinpoint physical events [5]. These data are highly indicative of personal preferences and behaviors thus making it possible to predict demographics (age, gender, income, etc.) of users through their analysis [6].

Hence, in this paper we aim to overcome the shortcomings of basing CF recommendation systems solely on the consumption of catalog items by enhancing these systems with Internet activity logs and Internet search engine logs. Our hypothesis is that the rich data representation enabled by this data can be leveraged to enhance CF based recommendation systems. Furthermore, this representation goes towards addressing the cold user problem, as even users who never consumed an item from the catalog, have their preferences encoded by the sites they visited, allowing the system to find other users with similar item preferences.

A. Background and Related Work

Recommendation systems technologies aim to serve relevant items from the catalog to a user based on the user's previous interaction with the catalog (e.g. rating or purchasing catalog items). There exist several families of approaches to the solution of these problems. The chronologically earliest of these is the content-based recommendation paradigm [7]. In this approach, features of catalog items, which are in general unstructured real world entities such as movies or games, are defined and encoded into a machine analyzable form. Using this representation, along with user's past interactions with the item catalog, a model of the user preferences is constructed which is capable of scoring all items in the catalog the user has not yet interacted with. Often TF-IDF vectors derived textual descriptions are used to create such a representation.

A alternate approach to the construction of recommendation systems is the CF paradigm [8]. The underlying intuition is to use the item catalog interactions of other users with "similar preferences" to select appropriate items to recommend. A key idea is that similarity between users can be inferred solely by their interaction behavior. Thus, the classical formulation of CF algorithms ignores all attributes of user information and make use only of the user/item matrix of interactions (e.g. ratings or purchases). This gives CF techniques better generalization across application domains, as, unlike content-based methods, there is no longer any need for defining and collecting metadata features to encode items and/or users.

Within the CF paradigm there are two main classes of algorithms. In the first of these classes, neighborhood methods [9], [10], use the implicit similarity of users/items to select catalog items to recommend to the user. Item based neighborhood methods [10] recommend items similar to other items the user has consumed. A user-based neighborhood method [9] recommends items rated highly by users similar to the user requesting the recommendation.

More recently, a class of approaches called Matrix Factorization (MF) (also known as latent factor modeling), has become very popular for CF. The key idea in MF models is to project both user and item representation into the same low-dimensional latent feature space. The rating that a particular user gives to a catalog item can then be approximated by the inner product of the corresponding user and item vectors in the latent feature space representation. The modeling assumption of an underlying low-dimensional subspace is motivated by the fact that users have affinities for a particular relatively small number of item features (e.g. genre, director, lead actor in the movie recommendation domain). Factorization of the user-item interaction matrix into the latent space representation described above has been widely explored in the literature [11], [12]. Due to the sparsity of this matrix, traditional methods for factorization such as Singular Value Decomposition (SVD) yields poor results. Recent works [13], [14] have posed this problem as a regularized regression of available user ratings (see section II).

CF based recommendation systems (specifically MF approaches) are easier to generalize across different recommendation domains than content-based approaches. They are also better for achieving serendipity, the recommendation of unexpected yet relevant catalog items. However, when presented with a "cold user", having no previous catalog interactions, it is unclear how to provide item recommendations in the CF framework. CF approaches also suffer from limited explainability of recommendations. In contrast, explanations for content-based method recommendations are more straightforward to derive.

Recent work by Li et al. [15] considers the problem of incorporating external signals from web activity as we do in this paper. Their method requires a great deal of meta-information on both the items and web-pages consumed by users in order to create quantitative features, making their approach at least partially content based. In contrast, the approach proposed in this work requires little knowledge of the domain beyond the user consumption information. This aspect of our proposal makes integration of additional meta-signals such as browsing and search history straightforward for entities that already have a recommendation system (especially one based on MF) in place.

B. Contributions

The main contributions of this work are:

- Novel, domain-independent and privacy preserving methods for enhancing MF models by expanding the user-item matrix and by imputation of the user-item matrix, using browsing logs and search query logs.
- Extensive experimentation with two real datasets from different domains, constituting different interaction signal types: *(1)* smartphone app purchases; and *(2)* online retail store item clicks. We present both quantitative empirical results and a qualitative anecdotal evaluation to support the merits of our proposed approach.
- Reports on negative results for the benefit of the community.

II. Matrix Factorization and Collaborative Filtering

We briefly introduce the basics of Matrix Factorization (MF). MF has established itself as a mainstream learning method for CF recommenders [16], [11]. MF-based methods are increasingly used in large-scale recommendation systems (e.g., Xbox [17], Azure Recommendations [18], Mahout [19]). Furthermore, MF-based models for CF have repeatedly demonstrated better accuracy than other methods such as nearest-neighbor models and restricted Boltzmann machines [20], [21].

MF algorithms are given a description of items and users as input, typically this description takes the form of a sparse vector, whose values are either explicit ratings or values inferred from the user data. The output is a set of vectors, one for each user and one for each item, spanning a *joint* user-item latent space. Future user-item interaction can be predicted based on vector space similarity between the corresponding vectors. Similarly, the similarity between any pair of catalog items is approximated by the vector space similarity of the corresponding pair of vectors.

The vectors are of dimensionality k; this value is given to the algorithm as a parameter. Each item i is represented by a vector v_i, and each user u is represented by a vector v_u. A large inner product $v_i^T v_u$, implies that item i is a good candidate for recommendation to user u. In order to learn the latent item and user vectors v_i and v_u, MF minimizes the squared error on the known values from the full user-item interaction matrix. Regularization terms (denoted λ below) are often added to reduce overfitting. The resulting optimization formulation is as follows:

$$\min_{v_u, v_i | u \in M, i \in N} \sum_{(u,i) \in K} (r_{ui} - v_u^T v_i)^2 + \lambda(|v_u|^2 + |v_i|^2)$$

where K denotes the set of all known usage points, M denotes the set of all users, N denotes the set of all items.

λ is tuned empirically. Solving the above minimization is equivalent to factorizing the original matrix into two matrices, one containing a k-dimensional vector for each user (*user* model), and another containing a k-dimensional vector for each item (*item* model). MF algorithms can be extended to use implicit signal and account for biases related to users or items, as discussed in detail in [16]. After the application of these extensions, the optimization problem takes the form:

$$\min_{b_u,b_i,v_u,v_i|u\in M,i\in N} \sum_{(u,i)\in K} (r_{ui} - \mu - b_i - b_u - v_u^T v_i)^2 + \lambda(|v_u|^2 + |v_i|^2 + b_u^2 + b_i^2)$$

Where K,M, and N are defined as above, μ denotes the average of all values of points in K, and b_i (b_u) denote the item (user) bias, in terms of difference from μ. This factorization process is illustrated in Figure 1, where P and Q are matrices composed from the output of the factorization of U, which is the user-item interaction matrix. The rows of matrix P are k-dimensional latent vectors each corresponding to a user. Similarly, the columns of Q correspond each to an item.

A. Matrix Factorization with Implicit One-Class Feedback

While much of the research into recommender systems is focused on numeric ratings, in most real-world systems, the available interaction signals are often limited to positive, unweighted and implicit feedback from users. Examples of this type of signal include: retail item purchase history, movie watching patterns and app download record. This type of signal is referred to as *one-class* data [22].

We require only one-class positive usage points for items and for the enhancing data. The baseline model that we enhance is a one-class MF model, presented in [17]. Basically, this model factorizes the one-class input matrix, and allows us to compute the probabilities p_{ui} – the probability that user u will interact with item i, using the inner products of the corresponding vectors.

This algorithm has been extensively optimized for scale and accuracy, may be extended to use weighted signals, when available, and is being used to serve recommendations to over 50 million Xbox users. It is also the default collaborative filtering algorithm used in the Microsoft Azure recommendation service [18].

B. Evaluating One-Class Algorithms

Evaluation of recommenders, even in its offline form, is a complex and multidimensional problem [23], [24]. In this work, we concentrate on evaluating prediction accuracy and its tradeoff with catalog coverage.

Accuracy. Offline accuracy of rating-based CF is typically measured with *Root Mean Squared Error* (RMSE) [24]. For one-class problems, where only implicit feedback is available, the commonly used metric for accuracy is *Mean Percentile Ranking* (MPR) [25], [26], defined as follows.

For every user item pair (u, i) in the test-set, we rank all items not in u's item interaction history and compute the percentile rank PR_{ui} of item i with regard to this ranking:

$$PR_{ui} \stackrel{\text{def}}{=} \frac{1}{|N| - |\Omega(u)|} \sum_{i' \notin \Omega(u)} \mathbb{I}\Big[p_{ui} < p_{ui'}\Big]$$

where N is the set of catalog items, $\mathbb{I}[\cdot]$ is the indicator function, $\Omega(u)$ are the items in u's item interaction history, and p_{ui} is the probability that user u will interact with item i. The MPR metric is computed by averaging the percentile ranks, PR_{ui}, over all test examples. Our experience with many datasets also shows a very tight correlation between MPR and the precision-at-k metric.

Coverage. Yet another important aspect of evaluating a recommender system is catalog coverage provided by the recommender [24]. That is, the percentage of catalog items that the recommendation system considers as candidates for recommendation to system users. The great promise of recommendation systems is allowing a larger portion of the item catalog to be surfaced to users. Coverage is a metric that quantifies the potential of achieving this promise. Coverage also affects other evaluation aspects such as novelty and serendipity. Pure CF algorithms cannot model cold items or users, creating a tradeoff between accuracy and coverage, particularly, for large dynamic commercial catalogs.

III. Methods

We present two approaches for recommendation enhancement. The first approach expands the user-item interaction matrix by describing each user as a vector which contains information on both item catalog interaction and browsing behavior. We call this method *matrix expansion*. Note that these signals are both one-class and positive (i.e., we do not have weights or any information on items for which the user has explicitly stated negative preference). The second approach adds interaction points inferred from the browsing logs to the user-item interaction matrix, which is typically very sparse [27]. The practice of adding external values to a matrix is usually referred to as *matrix imputation* [11].

A. Preliminaries

We shall make use of the following notations:

- M - the set of users;
- N - the set of items;
- U is an $|M| \times |N|$ matrix representing the set of known interaction points, e.g., apps downloaded, pages viewed, movies watched. A known interaction between a user and an item is represented as 1 in the corresponding entry. The remaining entries are unknown (denoted as 0 in U, for convenience).
- B - the set of *enhancing items*: URLs (truncated to include only the hostname) or search queries (stemmed);
- U_B is an $|M| \times |B|$ matrix representing the set of known enhancing interaction points, e.g. URLs visited or search queries made;

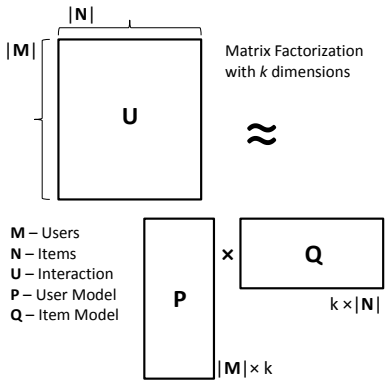

Fig. 1. Matrix Factorization of a matrix describing known interactions of users in M with items in N, into a joint space of dimensionality k.

Algorithm 1 EnhanceExpand(N,U,U_B)

```
for each row r in U, corresponding to user u do
    let r_B denote the row in U_B corresponding to u;
    w ← concatenate(r,r_B);
    add row w to matrix W;
end for
factorize W into (P, Q');
for each column q in Q', corresponding to item i do
    if i ∈ N then
        add column q to matrix Q;
    end if
end for
return (P, Q);
```

- $f_B : B \rightarrow N$ - the (possibly partial) imputation function.

B. Enhancing with Matrix Expansion

The typical input to a CF algorithm encodes each user as a vector in which every element represents a catalog item, and the element's value encodes the level of user-item interaction. These vectors are the rows of U. Our matrix expansion approach augments this initial user vector by representing each enhancing item as an additional element. A one-class, positive signal is given to every element if and only if the user interacted with the corresponding enhancing item. More precisely, every user u, is now represented by a vector of length $|N|+|B|$, denoted w_u, with value 1 (only) for elements representing known interactions. The result of this process is an expanded matrix, denoted W, whose rows are the user vectors $w_1, w_2, \ldots w_{|M|}$.

To this matrix W, we apply one-class MF as described in [17]. The factorization yields a matrix P whose rows are the latent space vectors, p_u, representing all users $u \in M$, and the matrix Q whose columns are the latent space vectors, q_i, representing all items and enhancing items $i \in U \cup B$. Since we are are only interested in recommending catalog items to system users we discard the latent representation of the enhancing items. The expansion method is given formally by Algorithm 1, and is illustrated in Figure 2.

Once the user and item representations are computed, recommendations can be retrieved as described in Section II.

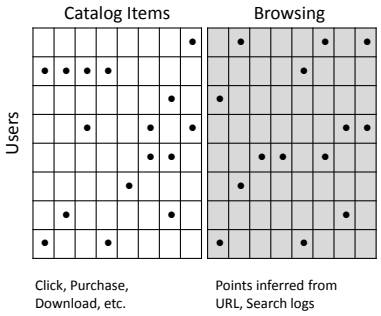

Fig. 2. Matrix Expansion: enhancing items, representing user behavior on the web, are used to expand the matrix.

The test set for computing accuracy (measured in MPR) does not include enhancing items. Since our algorithm does not consider enhancing items as candidates for recommendation to users, only items from the item catalog are ranked. For this reason, the comparison to models computed using only U is straightforward.

C. Enhancing with Matrix Imputation

The imputation enhancement approach we propose makes use of the set of of enhancing items (i.e., URLs and search terms) in a different way than the expansion enhancement approach described in the previous section. Rather than extending the classical CF user representation, the existing representation is densified with additional interaction points.

Each user representation is augmented by adding interaction points to catalog items, related to the enhancing items that the user interacted with on the web. More formally, let $R_u = \{i | i = f_B(b), f_B(b) \text{ is defined}, U_{B_{u,b}} = 1\}$, be the set of catalog items related to user interactions, via the imputation function f_B. We then represent every user u with a vector of length $|N|$, denoted w_u, which has 1 for all elements corresponding to items $i \in \{i | U_{u,i} = 1 \text{ or } i \in R_u\}$. We construct matrix W from these imputed vectors, whose rows are $w_1, w_2, \ldots w_{|M|}$. For each user $u \in M$ we create a representative vector p which is the average of all column vectors in Q, corresponding to items $\{i | i \in N, U_{u,i} = 1\}$.

We propose and experiment with the four imputation functions below, designed for URLs and queries as enhancing items. Each imputation function is described by two attributes, each of which takes one of two values:

1) ***all* vs. *last***:
   If *all*, we build a matrix where entry (i, u) represents the number of users who browsed URL u (or searched for query t) and interacted with catalog item i. If *last*, we order the URLs and items by their time of browsing or interaction, and build a matrix of items to URLs (or, queries), where the entry (i, u) is computed by counting the number of users who browsed URL u (or searched t) 10 URLs (or queries), or less, before interacting with item i. In both cases, for each URL u, we define: $f_B(u) = i$ (or $f_B(t) = i$) if u (or t) has the largest value in its column.
2) ***probs* vs. *no-prob***:
   If *no-prob*, we take the maximum from the original

Algorithm 2 EnhanceImpute(M,U,U_B,f_B)

for each row r in U, corresponding to user u **do**
  let r_B denote the row in U_B corresponding to u;
  $w \leftarrow r$;
  for each element e in r_B **do**
    let b denote the enhancing item corresponding to e;
    if $e = 1$ and $f_B(b)$ is defined **then**
      set the element in w, corr. to item $f_B(b)$, to 1;
    end if
  end for
  add row w to matrix W;
end for
factorize W into (P', Q);
for each user $u \in M$ **do**
  $G \leftarrow \{q_i | q_i$ is a column in Q s.t. $U_{u,i} = 1\}$;
  $p \leftarrow \frac{\sum_{q_i \in G} q_i}{|G|}$; //average the vectors in G
  add row p to matrix P;
end for
return (P, Q);

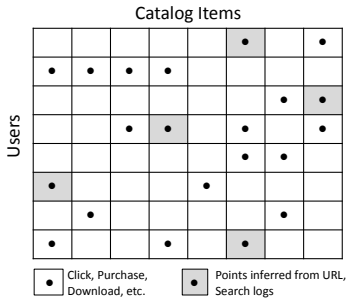

Fig. 3. Matrix Imputation. Points derived from browsing data reduce matrix sparseness.

matrices above.
If *probs*, we first normalize the count of each URL or query by the column sum of the matrix. Then, take the maximum. So, if a URL was visited by every user who bought an item, it will be scored lower than a URL that was only visited by people who interacted with a specific item.

The method of enhancement by imputation is given formally by Algorithm 2, and is illustrated in Figure 3. As in the previous method, retrieval and accuracy (MPR) are computed as usual. Again, since we consider only items from the catalog, the comparison of the imputation to models computed using only the set U is straightforward.

D. Privacy-Preserving Models

Most clients of recommendation systems (e.g., store operators) do not collect Internet browsing logs, nor do most collectors (and collators) of browsing logs have a need to provide recommendations to their users. However, for the methods proposed in this paper to be implemented successfully in a real world setting, these two sources of data must nevertheless be joined. This join must occur in such a way as to avoid compromising user privacy, which is of paramount importance among companies that handle large amount of personal data. In order to consider user privacy as preserved, either party must not be able to identify any user based on the data transferred to them by the other party. It is unlikely that any one user can be identified solely from their interaction with the item catalog of a single retailer, even if there is a large number of such interactions. In the absence of additional information such as demographics or location, uncovering the user's identity is a difficult task. Internet logs, even when limited to search engine queries, on the other hand, are known to be useful for identifying individual users [28].

This asymmetry in privacy-violating potential of the data held by the two parties in our scenario implies the correct flow of information in order to preserve privacy. This flow is for the store operator to provide the purchase list of anonymized customers to the browsing log collator, and have the collater create the recommendation model. In this case, some of the users of the store will be joined with their browsing logs through an identifier in the URL of the store, while for others only their purchase history will be available to the CF system when constructing the model. For the latter, the shop operator may have some information on websites visited by the users through referrer page information and cookie data.

Given a CF model built by the log collator, privacy can be retained even if the model is returned to the store operator because, given enough users and a small enough number of hidden factors (which is always the case in commercial systems where factors are 20-40 and users are several orders of magnitude more), individual user behavior cannot be deduced from the model itself, because the model represents the aggregated average behaviors of many users. Thus, the most privacy preserving method for data sharing is where the CF model is built by the log collator. This is the approach that we advocate in this work.

IV. Experimentation and Evaluation

This section presents experiments with two datasets collected from real users. Each dataset has two parts: catalog items interaction and enhancing item interaction. The enhancing item interactions consist of the record of visited URLs and search engine queries of users who gave their consent for their browsing behavior data to be anonymously collected. We note that URLs were truncated to include only the domain name, so as to achieve better generalization.

The *retail* dataset contains item clicks from an online apparel store, and browsing logs for the same users, collected during May 2013. The *apps* dataset contains downloads of Windows Phone 8 Apps and browsing logs for the same users, collected between September and November 2013 (inclusive). These datasets were intentionally selected to represent diverse domains: the retail dataset contains page *clicks* on pages of physical goods, while the apps dataset contains *purchases* of virtual multimedia goods. Our experimental results, thus, demonstrate the applicability of our enhancement methods across different types of user-item interaction types as well as varied product catalogs.

In order to maintain user privacy, all user identifiers in the data were first anonymized by hashing, before the authors had access to them. They were then aggregated prior to analysis,

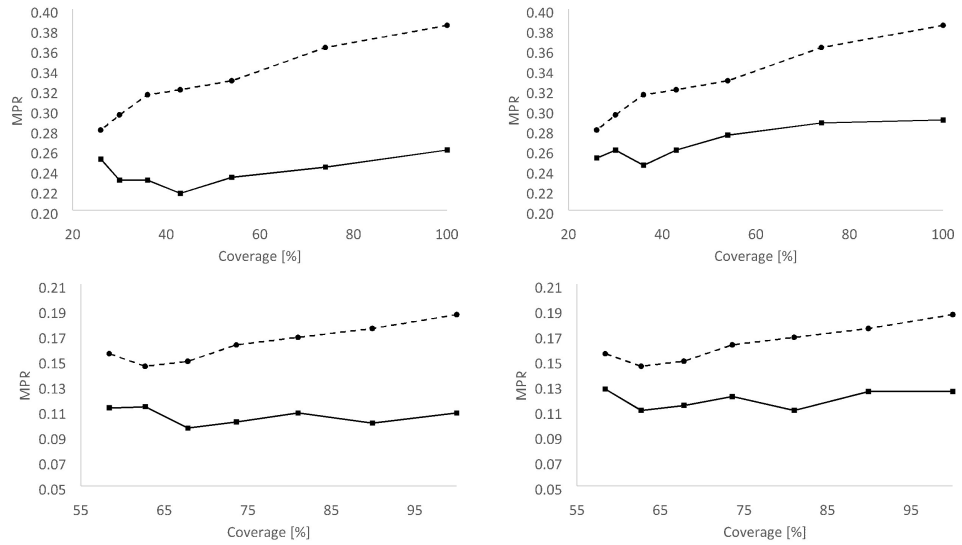

Fig. 4. Accuracy (as measured by MPR) as a function of coverage for matrix expansion with URLs. The top row shows results from the retail dataset, and the bottom from the apps dataset. The left column shows expansion using URLs and the right column using queries. Unenhanced values are shown in dashed line, and enhanced using a solid line. These results demonstrate the distinct improvement by the enhancement at all levels of catalog coverage. This plot also illustrates the tradeoff between accuracy and catalog coverage.

and no individual-level user datum was examined by the authors.

A. Data Description

Real world user-item interaction data typically follows a power-law distribution. Our data follows the power law as well. For example, following [29] for the user to URL part of the apps dataset, we have:

$$\frac{\mu}{|N|} = 0.000326 = \frac{\nu}{|M|}$$

Where M is the user set, N - items set, μ - average number of URLs per user, and ν - average number of users per URL.

The other datasets, of catalog items and of enhancing items, also follow power law distributions. Table 1 details the main attributes of our datasets. The support values for inclusion are 15 distinct users for a URL, and 5 for a query. The maximum values for inclusion are 400 distinct users for a URL, and 2000 for a query.

B. Experiment Setup

Our experiments examine model accuracy, using MPR (see Section II-B), as a function of the catalog coverage. From each of our two datasets, we derive 7 subsets, corresponding to 7 thresholds for catalog item inclusion. That is, at a given threshold only items having a number of interactions higher than the threshold value will be included in the model. These thresholds imply differing coverage values of the catalog, because, as the threshold is raised, fewer items have enough interactions to be included. Note that by this definition, the smaller the catalog coverage is, the warmer its items are (i.e., they have more interactions).

TABLE I. APPS DATASET AND RETAIL DATASET DETAILS.

Parameter	Retail	Apps
Users	48,362	137,181
Distinct Items	61,292	63,650
User-Item Points	349,290	2,815,509
Distinct URLs	79,286	81,977
URL Visits	8,661,815	3,330,998
Distinct Search Terms	16,079	106,877
Searches	421,786	1,462,985

For each such threshold (and for each enhancement method), we build models with and without enhancement, and compare their accuracy. The models were built using the Azure Recommendations service [18]. The test set includes a random sample of 1000 users. Following Section II-B, we hide one catalog item from the history of each test user, and check the rank given to it by the recommender.

Enhancing by Expansion. Figure 4 shows the accuracy as a function of model coverage for both methods and datasets. Enhancing by matrix expansion gives positive results for both datasets, whether using URLs or search queries as the enhancing items. The highest improvement in MPR was 72%.

Note that the expansion method can enhance a good model to be a very good model (apps), as well as enhance a fair model to be a good model (retail). This implies broad usability for enhancing recommendations with browsing data.

The results exhibit tradeoffs between accuracy and catalog coverage. Cold items are harder to model, and will harm model accuracy, as evident for models with high coverage values. However, as the experiments show, our methods are able to mitigate the accuracy loss for increased catalog coverage values.

Enhancing by Imputation. Enhancing by imputation with URLs gives positive and negative results, depending on the dataset and on the choice of f_B. For positive results,

TABLE II. RESULTS OF THE MATRIX IMPUTATION ON THE APPS DATASET AND THE RETAIL DATASET.

Imp. Method	Retail	Apps
URL, probs/last	Up to 24% (Fig. 5)	Negative
URL, probs/all	Up to 32% (Fig. 6)	Up to 14% (Fig. 7)
URL, no-prob/last	Negative	Negative
URL, no-prob/all	Negative	Negative
Queries	All Negative	All Negative

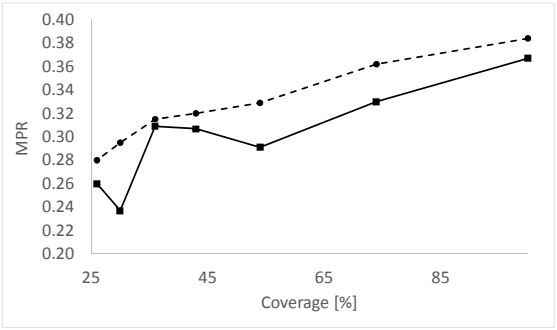

Fig. 5. Accuracy results for probs/last imputation with URL data for the retail dataset.

improvements are smaller than those of matrix expansion. However, imputation is computationally cheaper. Intuitively, this is due to the fact that the input matrix remains of the same dimension, and the number of points is much smaller than in expansion. Imputation with search queries did not yield positive results. Table 2 summarizes our findings for matrix imputation. Graphs 5, 6, 7 contain accuracy results for the successful methods. Here too, the results exhibit tradeoffs between accuracy and catalog coverage.

Modeling Query Topics. We also experimented with modeling topics of search queries. Classification to topics is done by a proprietary Bing classifier which relates each query with one of 63 categories, including, for example, commerce, tourism, weather-related and adult-themed queries.

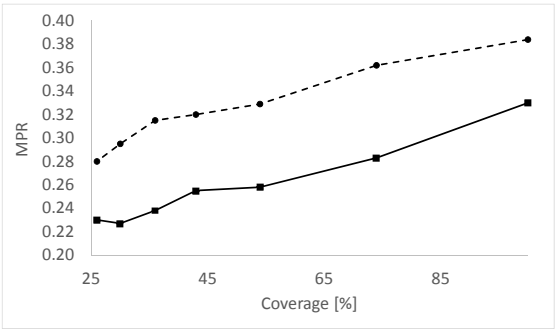

Fig. 6. Accuracy results for probs/all imputation with URL data for the retail dataset.

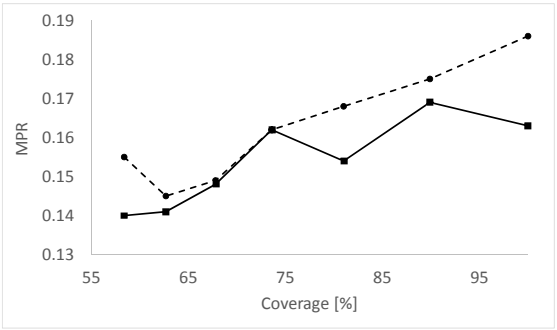

Fig. 7. Accuracy results for probs/all imputation with URL data for the apps dataset.

TABLE III. EXAMPLES FOR QUALITATIVE EVALUATION. APP 1 AND APP 2 ARE RECOMMENDED TO THE USER BASED ON A SINGLE VISIT TO THE URL IN THE LEFTMOST COLUMN.

URL	App 1	App 2
marriott.com *(hotel chain)*	Fly Delta *(airlines)*	Open Table *(restaurant reservation)*
lego.com *(kids toys)*	Hangman *(spelling game)*	Craftworld *(quest game)*
foodnetwork.com *(food portal)*	Couch to 5K *(fitness)*	Weight Watchers *(diet management)*
hulu.com *(tv and movies)*	AMC Theatres *(cinema chain)*	Pizza Hut *(pizza delivery)*

We expanded the user-item matrix (Algorithm 1) with topics signals. The resulting models did not show improvement over unenhanced models. We hypothesize that this is because the query topics are not detailed enough to distinguish between similar, though different, needs.

C. Qualitative Evaluation

To give intuition on the quality of prediction, and demonstrate the richness of Internet logs discussed in Section I, we show a few examples for qualitative evaluation. Each example consists of a URL, and two apps recommended solely based on similarity to the vector representing the URL. The model used to produce these examples is an expansion apps model (which includes URLs with 15 or more unique users). The sometimes amusing examples show instances of complementing services, demographic features and even a health tip, as detailed in Table III. Recommendations based on a single URL can be useful for predicting immediate needs or users' short-term intent.

V. DISCUSSION AND FUTURE WORK

Recommender systems suggest items to users by finding other users similar in their item preference. As such, they have become ubiquitous in Internet services and stores. To date, most recommender systems have defined similarity among users narrowly, bootstrapping the consumption of items to find similar users, and then to suggest future items. In this paper we propose to use an external data source, that of internet activity logs and search engine logs, to measure the similarity among users, as these data are known to describe many different facets

of people's behavior. Our approaches preserve the privacy of users while significantly enhancing the performance of MF-based CF.

Our results, tested on two very different catalogs, show that our proposed approach is a valuable addition to CF systems. The qualitative examples provide a hint as to why these recommendations work. When people visit a hotel's website, they are usually traveling. Hence, it is not surprising that they should want to download applications that are useful to travelers, such as those of restaurants or airlines. Our quantitative analysis confirms that our enhancement improves the accuracy of prediction, by as much as 72%.

The experimental results demonstrate that enhancing CF systems with browsing histories (URLs) is usually better than enhancing with search queries. This is probably because URLs contain a more detailed description of behavior. For example, a query is sometimes ambiguous, but the series of URLs that are browsed after the query can frequently resolve this ambiguity, thus helping to clarify the user intent. Thus, URLs are superior to queries or query topics, probably because their fine-grained resolution is needed in order to distinguish between the multitude of interests. Future work will attempt to integrate the two so as to ascertain if, and to what extent, they are redundant or complementing. Somewhat similarly, the results indicate that expansion methods perform better than the proposed imputation methods, probably because of a complexity-accuracy tradeoff. Thus, even though both models derive their input from the same data, CF systems benefit from the sparsity afforded by expansion.

Our results also indicate that models enhanced using Internet activity (both URLs and queries) are typically less sensitive to the existence of cold items than unenhanced models. This is expected since our enhancing data, to a certain extent, turns cold users into warm users, and turns cold items into warm ones. This is an additional benefit of the proposed enhancement, which alleviates one of the basic problems of CF systems. It should be noted though that the increase in recommendation accuracy persists also when considering only warm catalog items with many user interaction, thus the bootstrapping of cold items is not the only component contributing of the increase in accuracy. The semantic richness afforded by the many degrees of freedom represented in internet activity logs allows us to achieve an improved model of preference even for users with many catalog item interactions.

Finally, the reader will notice that across many of the experiments, for very low catalog coverage rates, MPR tends to slightly go up when coverage goes down. This is because after a certain point, when reducing coverage, informative items are removed, thus causing a reduction in accuracy.

In our work we have only used the fact that a user visited a URL as input to the CF system. We are currently investigating whether the contents from visited URLs or information about user engagement with the site can be incorporated into the expansion or imputation methods, to provide a richer user model, also based on contents. We are also investigating the dependence of model accuracy on the number of past interactions by users, to validate whether the system can make better predictions than traditional CF systems for users with few interactions ("cold users").

Learning to predict from one domain (e.g., music preferences) into another (e.g., movies) is a difficult problem which has received recent attention. In the future, we plan to leverage the informative power of browsing logs for cross-domain recommendations, where browsing-based user modeling serves as a meta-domain.

REFERENCES

[1] P. Resnick and H. R. Varian, "Recommender systems," *Communications of the ACM*, vol. 40, no. 3, pp. 56–58, 1997.

[2] S. Goel, A. Broder, E. Gabrilovich, and B. Pang, "Anatomy of the long tail: ordinary people with extraordinary tastes," in *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 2010, pp. 201–210.

[3] G. Eysenbach, "Infodemiology: tracking flu-related searches on the web for syndromic surveillance." in *AMIA... Annual Symposium proceedings/AMIA Symposium. AMIA Symposium*, 2005, pp. 244–248.

[4] Y. Ofran, O. Paltiel, D. Pelleg, J. M. Rowe, and E. Yom-Tov, "Patterns of information-seeking for cancer on the internet: an analysis of real world data," *PloS one*, vol. 7, no. 9, p. e45921, 2012.

[5] L. Backstrom, J. Kleinberg, R. Kumar, and J. Novak, "Spatial variation in search engine queries," in *Proceedings of the 17th international conference on World Wide Web*. ACM, 2008, pp. 357–366.

[6] S. Goel, J. M. Hofman, and M. I. Sirer, "Who does what on the web: A large-scale study of browsing behavior." in *ICWSM*, 2012.

[7] M. J. Pazzani and D. Billsus, "Content-based recommendation systems," in *The adaptive web*. Springer, 2007, pp. 325–341.

[8] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Communications of the ACM*, vol. 35, no. 12, pp. 61–70, 1992.

[9] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999, pp. 230–237.

[10] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*. ACM, 2001, pp. 285–295.

[11] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[12] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization." in *NIPS*, vol. 1, no. 1, 2007, pp. 2–1.

[13] R. Bell, Y. Koren, and C. Volinsky, "Modeling relationships at multiple scales to improve accuracy of large recommender systems," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2007, pp. 95–104.

[14] A. Paterek, "Improving regularized singular value decomposition for collaborative filtering," in *Proceedings of KDD cup and workshop*, vol. 2007, 2007, pp. 5–8.

[15] Y. Li, J. Hu, C. Zhai, and Y. Chen, "Improving one-class collaborative filtering by incorporating rich user information," in *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, ser. CIKM '10. New York, NY, USA: ACM, 2010, pp. 959–968. [Online]. Available: http://doi.acm.org/10.1145/1871437.1871559

[16] Y. Koren and R. Bell, "Advances in collaborative filtering," in *Recommender Systems Handbook*. Springer, 2011, pp. 145–186.

[17] U. Paquet and N. Koenigstein, "One-class collaborative filtering with random graphs," in *Proceedings of the 22nd international conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2013, pp. 999–1008.

[18] R. Ronen, N. Koenigstein, E. Ziklik, M. Sitruk, R. Yaari, and N. Haiby-Weiss, "Sage: recommender engine as a cloud service," in *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 2013, pp. 475–476.

[19] A. Mahout, "Scalable machine-learning and data-mining library," mahout.apache.org, 2012.

[20] R. M. Bell and Y. Koren, "Lessons from the netflix prize challenge," *ACM SIGKDD Explorations Newsletter*, vol. 9, no. 2, pp. 75–79, 2007.

[21] G. Dror, N. Koenigstein, Y. Koren, and M. Weimer, "The yahoo! music dataset and kdd-cup'11." *Journal of Machine Learning Research-Proceedings Track*, vol. 18, pp. 8–18, 2012.

[22] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang, "One-class collaborative filtering," in *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. IEEE, 2008, pp. 502–511.

[23] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 1, pp. 5–53, 2004.

[24] G. Shani and A. Gunawardana, "Evaluating recommendation systems," in *Recommender systems handbook*. Springer, 2011, pp. 257–297.

[25] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. IEEE, 2008, pp. 263–272.

[26] H. Steck, "Training and testing of recommender systems on data missing not at random," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 713–722.

[27] M. Zanker, A. Felfernig, and G. Friedrich, *Recommender systems: an introduction*. Cambridge University Press, 2011.

[28] A. Cooper, "A survey of query log privacy-enhancing techniques from a policy perspective," *ACM Transactions on the Web (TWEB)*, vol. 2, no. 4, p. 19, 2008.

[29] M. E. Newman, S. H. Strogatz, and D. J. Watts, "Random graphs with arbitrary degree distributions and their applications," *Physical Review E*, vol. 64, no. 2, p. 026118, 2001.