Subjective Similarity: Personalizing Alternative Item Recommendations

Tolga Könik
eBay Inc.
San Jose, CA,USA
tkonik@ebay.com

Rajyashree Mukherjee
eBay Inc.
San Jose, CA, USA
rmukherjee@ebay.com

Jayasimha Katukuri[1]
University of Louisiana
Lafayette, LA, USA
jaykatukuri@gmail.com

ABSTRACT

We present a new algorithm for recommending alternatives to a given item in an e-commerce setting. Our algorithm is an incremental improvement over an earlier system, which recommends similar items by first assigning the input item to clusters and then selecting best quality items within those clusters. The original algorithm does not consider the recent context and our new algorithm improves the earlier system by personalizing the recommendations to user intentions. The system measures user intention using the recent queries, which are used to determine the level of abstraction in similarity and relative importance of similarity dimensions. We show that user engagement increases when recommended item titles share more terms with most recent queries. Moreover, the new algorithm increases query coverage without sacrificing input item similarity and item quality.

Categories and Subject Descriptors

H.3.3 Information Search and Retrieval – *Information filtering*; *Clustering*.

Keywords

eCommerce, Recommender Systems, Personalization, Context-aware alternative item recommendations

1. INTRODUCTION

At e-commerce websites like *eBay.com*, the primary means of navigation involves user initiated activities like *browsing* a category, *searching* by entering keywords, or *filtering* search results by selecting values for a set of attributes the system provides. These actions often require users to understand the ontology organizing the inventory and guess keywords leading to desired results. This process can get frustrating for users who are not familiar with the content of the site and have only a general sense of what they are looking for.

Recommendations provide an alternative approach to assist users in accessing relevant content faster. Unlike search engines that aim to answer user-formulated queries, recommendation engines provide content without requiring direct user input. However, this makes finding relevant content more challenging because user interest must be guessed based on indirect information sources such as short-term session history, long-term user behavioral data, ontology the site uses to organize inventory, and the state of active inventory. As recommendation systems are gaining popularity with ever growing behavioral data and computational means to process them, research for specific recommendation circumstances becomes more important.

In this paper, we are focusing on *alternative item recommendations* where we assume that the user has engaged with a *seed item* and the recommender engine aims to provide other options to the user that she/he could consider as alternatives to the original item. For example in an e-commerce site like eBay, a suitable opportunity for alternatives recommendations is when the user have selected to view the details of an item, but before she/he has started a transaction to purchase it.

Alternative item recommendations are different than similar-taste recommendations (e.g. *people who like "tents" also like "bicycles"*), and complementary item recommendations (*people who bought "iphone" may also buy an "iphone case"*) which are typically tackled with methods like collaborative filtering [1] because these methods can recommend items that do not serve as replacement for the original item. In contrast, alternative items should bare similarity to the original item at some level of abstraction. On the other hand, content-similarity based methods alone (e.g. *you lost the bid on this item, but we found another very similar one* [8]) may not be entirely sufficient to tackle this problem either since they may not provide sufficiently diverse options to the user.

We believe that alternative item recommendations should negotiate a trade-off between seed-item-similarity, diversity, and quality. To act as a replacement, the recommendations should have sufficient similarity to the seed item, yet differ in some dimensions (i.e. different price, color, capacity, etc.) to provide the user with new options. Furthermore, the recommended items should be potentially better than the seed item in some aspects (i.e. better price, better seller quality, better condition etc.) since otherwise the user would not have a reason to prefer them over the seed item.

The central claim in this paper is that the level of similarity needed for alternative recommendations is subjective and relative to the intention of the user. For example, if a user is shopping for an electronics gift item, a smart phone device can be a viable alternative to a tablet device. On the other hand, if the user intention is to buy an ultra-light laptop of a particular brand, the alternatives should have a closer similarity to the seed item. Therefore, we state that capturing information about users' intentions is key for improving alternative recommendations. Personalized recommendation systems often build models on user

WWW'15 Companion, May 18–22, 2015, Florence, Italy.
ACM 978-1-4503-3473-0/15/05.
http://dx.doi.org/10.1145/2740908.2741999

[1] work done while working at eBay Inc.

profile/taste/preference models [3] using long-term behavioral data. While we believe that interpreting short-term intensions can benefit from such models, the system we discuss in this paper takes the other extreme and utilizes only short-term behavioral data to predict intention and personalize recommendations.

In this paper, we describe an architecture that personalizes alternatives recommendations using short-term in-session data. In particular, we propose a method to determine the scope of items the user may be interested in and our architecture considers only recommendation candidates that are within that scope. Our system

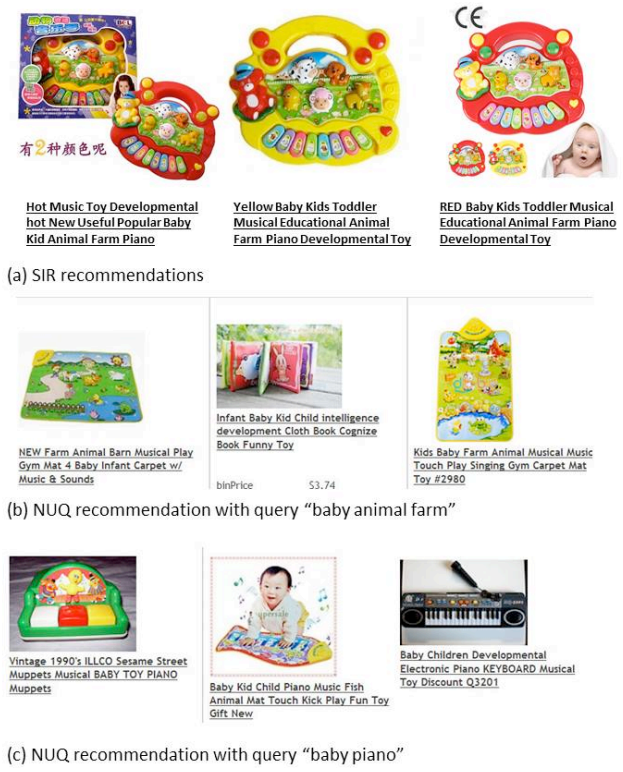

Figure 1. Recommendations by SIR and NUQ

also detects item traits in recent user queries and prefers items that contain those when selecting the top few recommendations from the pool of candidates. We hypothesize that personalizing recommendations using recent user queries improves utility of the recommendation system compared to a similar system that does not utilize that information.

We implemented the architecture described in this paper and evaluated it using data collected on the *eBay.com* site. At the time of evaluation, eBay was using an alternative recommendations algorithm that negotiates a trade-off between similarity to a seed item and quality of recommended items [8] but its recommendations are not contextualized to the actions of an individual user. We analyze data generated by this system and show that user engagement is higher when the recommendations are consistent with "scope of interest" as measured by our new algorithm. In the next section, we present a motivating example. Next, we describe our architecture followed by our evaluation and concluding remarks.

2. A MOTIVATING EXAMPLE

An opportunity for alternative item recommendations is when a user is looking at details of an item, but has not started the purchase process yet. At this point, the website may recommend alternatives to the current item to provide user with new options. SIR [8], a recommendation engine used in production at *eBay.com* in this kind of placement aims similarity to the seed item, but provides a customizable parameter to increase quality and diversity of recommended items by allowing reduction of similarity. The output of this algorithm does not depend on actions of the user prior to visiting the item page. In contrast, NUQ, the algorithm we describe in this paper, personalizes similarity by utilizing the query leading the user to the item details page.

Figure 1 shows recommendations by SIR and NUQ for a seed item titled "*New Useful Popular Baby Kid Animal Farm Piano Music Toy Development Hot*". The input item has multiple features and the challenge is to determine what dimensions the similarity should be based on and consequently, how much diversity will be allowed in the recommendations.

SIR (Figure 1.a) returns items similar to the seed item and it generates that recommendation independent of the query that leads the user to the item detail page. On the other hand, the recommendations of NUQ depend on the user query. In the first case (Figure 1.b), the system prefers baby toys that have an *animal farm* theme to satisfy the query but it also ends up retrieving items that are *developmental musical toys* to increase similarity to the seed item. When the input query leading to this item is "baby piano" (Figure 1.c), the resulting impressions also change dramatically, this time returning items that are pianos for babies, while trying to satisfy item similarity, i.e. in "developmental toy" dimension.

Both SIR and NUQ aim to find items similar to the seed item, but while the level of abstraction in similarity is controlled in SIR with a global parameter, in NUQ, the level of similarity and relative importance of similarity dimensions are gauged by user intention, for which the system uses user queries as an operational proxy. For example in Figure 1.b the dimension "animal farm" is prioritized over "piano" and similarity is abstracted by ignoring "piano" feature.

3. BACKGROUND: SIR ENGINE

The alternatives recommendation architecture we describe in this paper is based on SIR, a large-scale similar item recommendation engine that generated statistically significant business impact at eBay marketplaces after wide deployment compared to a naïve IR system [8]. The algorithm is efficient enough to cover hundreds of millions of items while serving tens of millions of active users.

The core idea in SIR is learning cluster expressions with massive offline processing and using those expressions to increase relevance, quality and efficiency of online recommendations. In the first step of the runtime architecture of SIR (Figure 2), the system inputs a seed item and retrieves a small set of clusters that match to that item using an in memory cluster dictionary. Next, it uses those clusters to construct a search query, which it utilizes to retrieve a set of recommended items from a large inventory of items.

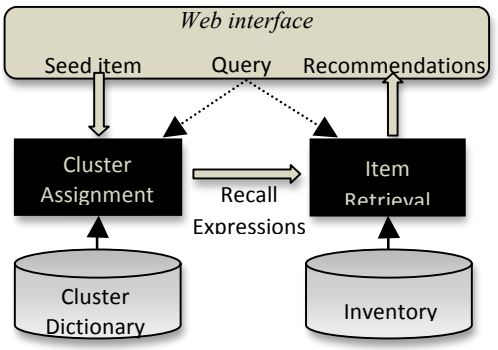

Figure 2. High-level architecture of SIR and NUQ. Dotted arrows are only part of NUQ.

3.1 Cluster Expressions

The SIR clusters are learned from a mixture of information sources including user queries, items in the inventory, user interaction with search filters. We will not describe that process here in detail because even though our work uses SIR clusters, it does not contribute how those clusters are learned. However, one important characteristic of SIR clusters is relevant for our discussion: SIR is biased to create clusters that group items that users view together in their search queries and therefore act as high-level features that represent subjective similarity. In particular, all SIR clusters are specializations of frequent user queries. Moreover, during constructing of clusters, SIR also uses data about search filters (i.e. *color=red*) people interact with in the search UI after entering a keyword query. Attributes associated stronger with a query *q* in that data set, are more like to be added to the cluster expressions that are constructed as specializations of *q*.

The cluster expressions are explicit definitions that group a set of items. SIR clusters consist of bag of phrases, categories the item belongs to, and set of attribute value pairs that describe items. Table 1 shows example clusters, where each cluster label is linked to a bag of features and Table 2 shows the items that correspond to these clusters. Note that, attribute features (e.g. *brand=Fisher Price*) and category features (e.g. *category=Baby Toys*) are not reflected in these examples to simplify the discussion.

Table 1. Cluster Expressions

Clusters
c_1: {baby, piano }
c_2: { musical, animal, farm, toy }

Table 2. Items and The Assigned Clusters

Item	**Title**	**Cluster(s)**
i_1	Childs Grand Baby Piano with Kids Bench of Solid Wood Construction	c_1
i_2	22 Lot FISHER PRICE Little People Musical Sound Farm Barn Animals Tractor CLEAN	c_2
i_3	RED Baby Kids Toddler Musical Educational Animal Farm Piano Developmental Toy	c_1, c_2

3.2 Recall Expression

SIR run-time engine first maps an input item to a small set of best matching cluster expressions, which it uses to construct a recall expression. The system uses the recall expression to retrieve a set of candidate recommended items.

The system uses the cluster model to determine what dimensions are important for similarity to a particular item. The cluster expressions are stored in a Lucene index[2] that retrieves clusters given a seed item's features. The system takes a normalized vector (similar representation with cluster expressions) generated from a seed item, and retrieves *n* clusters that best match that item. Typically, the seed item contains tokens missing in the retrieved clusters, but the system often retrieves clusters that are fully contained in the seed item vector. For example the item i_3 in Table 2 could retrieve clusters c_1 and c_2 in Table 1. The system is also capable of fuzzy matching and retrieves clusters that maximize inclusion of important terms in the seed item. The term importance is determined by the rareness of the term in the model.

In the next step, the system selects the top *n* clusters and generates a search query using the tokens shared between the seed item and clusters. If for example, the matching tokens are $\{t_{11}, t_{12}, \ldots\}$ and $\{t_{21}, t_{22}, \ldots\}$, where t_{ij} is the *j*th matching token of the *i*th best matching cluster, we construct the recall constraint expression:

$$(t_{11} \text{ and } t_{12} \text{ and } \ldots) \text{ or } (t_{21} \text{ and } t_{22} \text{ and } \ldots) \text{ or } \ldots$$

Each conjunction in this expression retrieves items similar to the seed item in some general dimension. Moreover, further refinement on similarity is left to the next phase, which negotiates a trade-off between similarity and quality.

3.3 Item Ranking

In the next phase, the system retrieves a small ordered set of items, which are ranked by a tradeoff between quality and seed-item-similarity:

$$Score(seed, reco) = w_1 * Sim(seed, reco) + w_2 * Quality(reco)$$

Here, *seed* is the seed item and *reco* is the candidate item for recommendation. The ranking score is determined by a weighted average between a similarity function *Sim* and quality of the recommended item as measured by a *Quality* function. Even though calculating *Sim* and *Quality* functions can be complicated, SIR can scale the trade-offs between them to a very large volume of traffic because this calculation is conducted only on a small set of items retrieved with the recall expression rather than the whole inventory. SIR uses a similarity function *Sim* that compares the shared and not shared tokens between *seed* and *reco*, but weights the terms with domain specific importance [8]. The details of the Quality function are less important for our discussion.

4. NUQ ARCHITECTURE

In this section, we describe our personalized alternative item recommendation architecture NUQ (Near User Queries). NUQ is a run-time system that takes a seed item and a set of user search

[2] http://lucene.apache.org/core/4_10_1/index.html

queries as input and returns a small set of items that are similar to the seed item and are consistent with the user queries.

At high-level, it works in two main phases that are similar to SIR's main phases (Figure 2). In the first phase, NUQ assigns the seed item to clusters, creates recall expressions using those clusters, and in the second phase, it orders items that satisfy those expressions. However, the details of these phases differ in an important aspect: Both phases accept past user queries as an additional input and skew the results to maintain consistency with those. Consequently, NUQ customizes its recommendation closer to the short-term user intentions by preferring items that are consistent with recent queries.

4.1 Boosting Cluster Assignment

The cluster assignment phase starts by retrieving raw item features. In our use case, cluster assignment uses item title and category id, and item attributes as raw data. The category id refers to the ontology that organizes all items in the inventory into a hierarchy. The item title goes through a number of normalization steps identical to SIR [8] including spell correction, normalization, feature extraction, which results in an *item feature vector* consisting of a set of normalized phrases and feature value pairs.

Unlike SIR, NUQ cluster assignment also inputs a query and runs it through the same normalization steps to create a *query feature vector*. Next, the algorithm selects a set of features in the query vector that are consistent with the item vector and marks them as *context boosting factors (CBF)*. The algorithm aims to prioritize recommendations that are consistent with these boosting factors. Our current implementation creates CBF only using the last query, but a natural generalization is to use queries in the recent history and weight the effect of factors by recency.

In cluster assignment step, like SIR, NUQ returns cluster expressions similar to the item vectors utilizing a TF-IDF measure, which prioritizes rare features over more frequent ones. However, unlike SIR, NUQ boosts item features that are consistent with the query vector such that given a pair of clusters C_1 and C_2, C_1 is guaranteed to rank lower than C_2, if $C_1 \cap CBF$ is a subset of $C_2 \cap$ CBF.

For example if the last user query is "baby piano", the system is guaranteed to prioritize cluster that contain both of the terms "baby" and "piano", if any such cluster exists. Moreover, any cluster that contains "baby" or "piano", is also guaranteed to be prioritized over clusters that do not have any of those terms. Finally, the relative importance of a cluster that contains "baby" (but not "piano"), over another cluster that contains "piano" (but not "baby"), is determined by the rareness of those terms in the cluster dictionary.

Once the clusters are retrieved, a recall expression is generated similar to SIR as described in section 3.2, which is next used to retrieve a set of items from the inventory.

4.2 Boosting Item Ranking

In the second phase, like SIR, NUQ orders its results to select top few recommended items. However, unlike SIR ordering function that negotiates between seed item similarity and quality, NUQ also prefers items that have terms consistent with context boosting factors obtained from queries. In particular, the *Sim* function describes in section 3.3, utilizes an importance weight for each term. In NUQ, these weights are boosted with a large number if the term belongs to the user query.

If all clusters NUQ retrieves in the first phase contain all boosted query terms, all items that match the recall expression would contain all query terms. In that circumstance, boosting at ranking time would not have any utility because all retrieved items would contain these query terms and boosting would not have an effect on ranking. However, NUQ can retrieve clusters with missing query terms. This can happen when no cluster expression contain all of these query terms, for example when the query is a very specific one. In that case, it is still useful to try to satisfy user query terms in ranking time. SIR ranking already employs an algorithm that measures seed item similarity using weighted terms. NUQ only changes this scheme by boosting the weight of user query terms that are in CBF set.

5. EVALUATION

We are hypothesizing that increasing relevance with respect to past queries increases relevance for users in alternative items recommendations. To evaluate this hypothesis we analyzed how users are interacting with the SIR system in an alternatives recommendation setting (prepurchase similar item recommendations on an item details page). We collected data when users search a query and enter item details page (seed item) and see impressions generated by SIR based on similarity to the item on the page. We show that recommendations consistent with query terms have stronger engagement and NUQ generates (as expected) recommendations more consistent with user queries.

Normally, the seed item is consistent with the query terms, since the item is returned as a search result for the query. Exceptions can happen due to unusual navigation patterns or tracking errors and we kept them out of our analysis. We determined that 65% of recommendations do not contain all query terms even when the seed item contains 100% of query terms. This suggests that there is potential room for improvement if query term boosting is improving engagement.

Next, we analyzed engagement on recommendations with respect to the number query terms they cover (Figure 3). Here the horizontal axis represents percentage of query terms the recommendations cover and the vertical axis shows the difference in click-through rate (CTR) (we report only relative percentage difference with respect to the average CTR in that placement because we are not allowed to publish absolute CTR values). This supports our claim that users are finding recommendations consistent with the query more engaging. Recommendations that contain all query terms have 60% higher CTR compared to recommendations that do not contain query terms.

Next we investigated to what extent NUQ generates recommendations with better query coverage. We ran NUQ and SIR on a data set consisting of user queries and the items users viewed after these queries. We discovered that NUQ recommendations increase query term coverage by 43% over SIR. We have also measured similarity of the recommended items with the seed item between NUQ and SIR (measured by Jaccard similarity of titles) and found out that the difference is very small. Moreover, we compared the average quality of the recommend items as measured by an internal metric used in several eBay systems. NUQ was on par with SIR on item quality as well. As a result, NUQ is increasing query similarity, without sacrificing seed item similarity and quality.

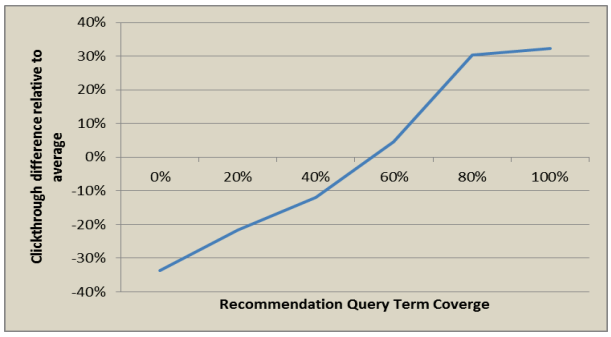

Figure 3. CTR increases with recommendation query coverage

Since seed item title contains all query terms, increase in similarity between seed item and recommendations can indirectly increase query term coverage of recommendations. To claim that CTR is increasing with query term coverage, we need to control for similarity between seed item and recommended items. Figure 4 shows change of CTR with respect to query term coverage for fixed seed-recommendation item similarity values. CTR's are again reported as percentage differences with respect to mean CTR of this placement. In this graph, we observe that CTR increases with query term coverage when seed item similarity is kept fixed.

In summary, we showed that SIR performs better in recommendations that include more query terms and NUQ generates recommendations with more query terms without sacrificing quality of seed item similarity. While this does not guarantee that NUQ will increase engagement or financial impact, it suggests that personalizing similarity recommendations with in-session query information is a promising direction to explore better alternative recommendations systems.

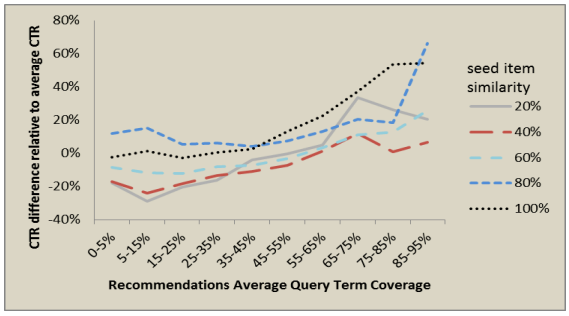

Figure 4. CTR vs. query coverage for fixed seed similarity

6. RELATED WORK

Large scale recommendation engines like Amazon's product recommendations [1], YouTube video recommender system [2], and Google news personalization service [3] are popular and used routinely by a large volume of users.

Anand and Mobasher [5] addresses the problem of incorporating context within recommendation systems. The paper distinguishes user's short-term and long-term memories and defines a recommendation process that uses both of them. Our proposed work utilizes the user's short-term contextual information. Some version of Netflix movie recommendations [4] utilizes the user's long-term activity. Baltrunas et al. [6] introduces context related factors into matrix factorization for item rating predictions.

Most of the existing recommender systems address recommendations in a stable collection of items or products. Amazon's recommender system [1] works in the space of products that are stable and do not expire in a short period time. Netflix recommends movies [7] from a slowly growing collection. Therefore, both of these systems can pre-compute item-item relationships using collaborative filtering methods. The Google news personalization [3] is one of the few works that addresses the issue of recommendations when there is item-churn.

7. CONCLUSION

In this paper, we outlined NUQ, an algorithm to extend an existing similar item recommendation system SIR to make context sensitive alternative item recommendations. The new algorithm boosts its recommendations using user queries. We analyzed SIR recommendations with real transactional data and determined that it is performing better when the recommendations are consistent with the query. Moreover, we showed that NUQ generates significantly higher query coverage without sacrificing seed item similarity, which suggest that NUQ may outperform SIR in user engagement.

Our current report is preliminary but it motives use of queries to personalize alternative item recommendations. Future work involves finding the right balance to boost query terms to obtain performance gain over SIR on live site traffic.

REFERENCES

[1] Linden, G., B. Smith, and J. York. 2003, Amazon.com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing*. 7, 1, 76-80.

[2] Davidson, J., Liebald, B., Liu, J., Nandy, P., Van Vleet, T., Gargi, U., Gupta, S., He, Y., Lambert, M. Livingston, B., and Sampath, D. 2010. The YouTube video recommendation system. In *Proceedings of the Fourth ACM Conference on Recommender Systems* (Barcelona, Spain). ACM, New York, NY, 293-296.

[3] Das, A. S., Datar, M., and Garg, A. 2007. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th International Conference on World Wide Web* (Banff, Alberta, Canada). ACM, New York, NY, 271-280.

[4] Amatriain, X. 2012. Mining large streams of user data for Personalized Recommendations. *SIGKDD Explorations*.14,2.

[5] Anand, S. S. and Mobasher, B. 2007. Contextual recommendation. In *From Web to Social Web: Discovering and Deploying User and Content Profiles*. Springer-Verlag, Berlin, Heidelberg, 142-160.

[6] Baltrunas, L., Ludwig, B., and Ricci, F. 2011. Matrix factorization techniques for context aware recommendation. In *Proceedings of the fifth ACM conference on Recommender systems*. ACM, New York, NY, 301-304.

[7] Koren, Y. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Las Vegas, Nevada). ACM, New York, NY, 426-434.

[8] Katukuri, J., Könik, T., Mukherjee, R., and Kolay, S. 2014. Recommending similar items in large-scale online marketplaces. In *2014 IEEE International Conference on Big Data*, 868-876.