Improving Product Search with Session Re-Rank

a CS341 data mining project

Charles Celerier (cceleri), Bill Chickering (bchick), and Jamie Irvine (jirvine)

June 9, 2013

Walmart.com maintains an online catalog of over 2M products. Consequently, enabling users to quickly find products that conform to their specific needs and tastes is especially challenging. Given the difficulty of its task, Walmart.com's product search engine does an impressive job of interpretting the user-provided query and rapidly returning relevant results. Yet, there remains highly significant information that is not fully leveraged. The details of a user's online shopping session are indicative of a user's intent and compliment—indeed, provide context for—the user-provided query. In this report we describe and analyze an ordering scheme we call *Session Re-Rank* (*SRR*) that can potentially induce a large increase in both click-through-rates and conversions on the first page of query results.

1 The Technique

SRR works by comparing previously clicked items with the top N items returned by the search engine in reponse to a query. Items in the top N that are sufficiently similar to previously clicked items are promoted. The extent (i.e. number of positions) of the promotion for a particular item is a function of its similarity to previously clicked items, its original position, and the promotions of other items.

The similarity between an item to be shown and a previously clicked item is determined within five distinct vector spaces: *click-space*, *cart-space*, *query-space*, *title-space*, *item-space*. The non-unique representation of an item within each of these spaces may be thought of as a binary vector or a set of objects. (MapReduce jobs process historic query data to construct indexes whose keys are itemids and values are lists of the appropriate objects. Great care went into ensuring that index entries can be accessed in $\mathcal{O}(1)$ and that two entries can be merged to compute their intersection or union in linear time.) The similarity $J_s(A, B)$ of two items, A and B, within a particular space s is determined using Jaccard similarity. Similarities within particular spaces are then weighted and summed to determine the composite similarity

$$S(A, B) = \sum_s C_s \cdot (J_s(A, B))^{\alpha_s}, \tag{1}$$

where C_s and α_s are tuning parameters. The score σ attributed to an item to be shown is then the summation of composite similarities between itself and all previously clicked items plus the click-through-rate (CTR) Γ_i of the item's original position i

$$\sigma(A) = \sum_{B \in P} S(A, B) + \Gamma_i, \tag{2}$$

where P is the set of previously clicked items.

Another important parameter of *SRR* is the insert position I_0, which indicates that the top I_0 positions of the original ordering are to remain fixed. For the results discussed in this report, we use $I_0 = 2$, meaning that we never reorder the first two items of query results. We found this configuration maximized our metrics, although, we suspect this may largely be due to users' bias toward clicking on the first one or two positions indepedent of what is shown there. That is, $I_0 = 0$ might prove optimal for an online implementation.

2 Similarity Spaces

The premise behind *click-space* is that two items are similar if they are both clicked within the same online shopping session. The dimensions, or objects, of this space are therefore past user-sessions. The *clicks-index*

for the data presented in this report was contructed using approximately half of the provided data, or about 60M queries (about 120M page views).

Cart-space is based on the notion that two items are similar if they ever appear in a shopping cart together. The objects of this space are therefore shopping carts. The *clicks-index* for the data presented in this report was also contructed using approximately half of the provided data.

Items are also considered similar if they appear in a query together. The objects of *query-space* are therefore queries. We make a distinction, however, between *user-queries* and *unique-queries*. The former are the well-defined entities within the raw Walmart data. The latter is an abstraction based on the notion that multiple *user-queries* can correspond to a single *unique-query*. To derive *unique-queries* from our data, we cluster *user-queries* according to following policy: two *user-queries* with the same search attributes (e.g. category or price filters) are considered the same *unique-query* if the strings constructed by concatenating the space-seperated, stemmed (we use the Python stemming.porter2 module), and forced to lower-case terms from each of their rawqueries are equal. We point out that while we achieved better results with this policy compared to simply using *user-queries*, we have no reason to believe that this is the ideal way to cluster queries for use within *SRR*. Indeed, we believe one way to improve *SRR* is to optimize the query clustering policy.

Title-space is straightfoward. Each item is associated with a set of terms from its title. We ignore case, but at present do not stem, discard stop words, or weigh terms in any way.

Finally, the structure of *item-space* is unique because it involves a level of indirection. The premise here is that if items A and B are clicked in a single user-session and items A and C are clicked in another user-session, that items B and C are similar because they have item A in common. In this way, a large number of relationships between items is created. *Item-space* resembles *click-space* in that if two items are clicked during a single session, they will have nonzero similarity. It differs from *click-space* in two key respects, however. First, items that have historically never been clicked in the same session can have nonzero similarity if they were each clicked with a common third item. Second, if items are clicked together in many sessions this will increase their Jaccard similarity in *click-space* but not in *item-space*.

3 An Example

To illustrate the efficacy of our technique, we present a real query example. The only fictitious part of the example will be our shopper's name, David. David is interested in the "Primo Ceramic Crock Water Cooler with Stand" and clicks on this item during his session. Sometime later he navigates to the "Grocery →Beverages →Water" category and searches for "water". He is presented with 300+ results and clicks the 90th item, a 3 liter jug of water.

The top six original results are compared to the *SRR* results in Table 1 where the first and third column represent the original ordering presented to David. Note that we reorder the 90th item from the original results to be the 3rd item in the *SRR* results. Tables 2 and 3 show the similarity scores from each index for each item in the two orderings compared to David's lone previously clicked item. In each of these tables, the first column correponds to Table 1. The remaining columns are calculated from Equations (1) and (2) using optimal tuning parameters.

We will show the calculation for the *item-space* similarity for the 90th item in the original ordering. By referring to the corresponding index for *item-space*, we can recall that the 90th item was clicked in a same session with 39 different items and the previously clicked item was clicked in a same session with 455 different items. The titles of the 13 items found in common are:

Great Value: Distilled Water, 1 Gal

Nestle Waters Bottled Spring Water, 24ct

Primo Mineral Water, 5 gal

Deer Park Sumo Bottle Natural Spring Water, 3l

Arrowhead Mountain Spring Water, 3l

PUR Advanced Faucet Water Filter Vertical - Chrome

Ozarka Natural Spring Water

Formula 409 All Purpose Lemon Scented Cleaner, 32 fl oz

	Original Ordering		***SRR* Ordering**
1.	Great Value Purified Water, 24ct	1.	Great Value Purified Water, 24ct
2.	Nestle Waters Bottled Spring Water, 24ct	2.	Nestle Waters Bottled Spring Water, 24ct
3.	Voss Water, 16.9 oz (Pack of 24)	90.	Arrowhead Mountain Spring Water, 3 l
4.	CA Cherry Sparkling Water, 1 l, 12pk	63.	Great Value: Distilled Water, 1 Gal
5.	CA Water, 1 l, 12ct	38.	Arrowhead Mountain Spring Water, 2.5gal
6.	CA Peach Sparkling Water, 1 l, 12ct	8.	CA Orange Sparkling Water, 1 l, 12pk

Table 1: Original Ordering vs. *SRR* Ordering for "water" query

	σ	**CTR**	**Clicks**	**Items**	**Carts**	**Queries**	**Titles**
1.	0.943 75	0.075 40	0.525 00	0.160 48	0.000 00	0.153 49	0.029 37
2.	1.202 11	0.039 00	0.686 16	0.210 10	0.000 00	0.239 45	0.027 40
3.	0.357 06	0.025 40	0.000 00	0.194 19	0.000 00	0.111 77	0.025 70
4.	0.752 68	0.019 50	0.451 34	0.198 29	0.000 00	0.060 63	0.022 92
5.	0.736 73	0.015 30	0.457 94	0.146 90	0.000 00	0.093 68	0.022 92
6.	0.174 83	0.012 90	0.000 00	0.096 78	0.000 00	0.042 24	0.022 92

Table 2: Index similarity scores of the top six original results to "Primo Ceramic Crock Water Cooler with Stand"

	σ	**CTR**	**Clicks**	**Items**	**Carts**	**Queries**	**Titles**
1.	0.944 00	0.075 40	0.525 00	0.160 00	0.000 00	0.153 00	0.029 47
2.	1.200 00	0.039 00	0.686 00	0.210 00	0.000 00	0.239 00	0.027 43
90.	0.953 00	0.000 23	0.657 00	0.223 00	0.000 00	0.045 50	0.027 44
63.	0.950 00	0.000 49	0.562 00	0.266 00	0.000 00	0.093 20	0.027 46
38.	0.938 00	0.000 91	0.645 00	0.262 00	0.000 00	0.000 00	0.029 40
8.	0.897 00	0.010 00	0.656 00	0.208 00	0.000 00	0.000 00	0.022 90

Table 3: Index similarity scores of the top six *SRR* results to "Primo Ceramic Crock Water Cooler with Stand"

Great Value Spring Water, 1 gal

Nestle Pure Life Purified Water, .5l, 35pk

Arrowhead Mountain Spring Water, 2.5gal

Primo Ceramic Crock Water Cooler with Stand

Augason Farms Emergency Water Storage Kit

We then calculate the similarity of the 90th item and the previously clicked item in *item-space* to be $13/(455+39-13) = 0.0270$. The corresponding score shown in Table 3 was calculated from the optimal tuning parameters.

We believe this example illustrates how our technique can form subtle relationships between items based on historical user session data. In this case, we have a shopper who had an interest in a water cooler and subsequently made a query for "water". If the shopper had been in a brick-and-mortar store, he likely would have browsed water jugs to use with the cooler he had picked up. In this case, *SRR* recognized David's previous click on a water cooler, related that water cooler to large water jugs, and showed David the water jugs he was looking for all along.

4 The Data

Walmart.com has generously supplied us with a large dataset consisting of about 250M pageviews comprising about 120M query results which occurred over about 30 days. The data includes the user-provided rawqueries together with search attributes, visitorIds and sessionIds, shown items, clicked items, which items were placed in a shopping cart, and which items were ultimately purchased. In addition, they have provided detailed item information including title, description, category, and other details. The query data was randomized with respect to search time and then segregated into three disjoint sets. The first set, which consists of about half of the data, was re-structured into indexes that form four of the similarity spaces (*click-space*, *cart-space*, *query-space*, and *item-space*) we use to identify relationships between items in realtime (the remaining similarity space, *title-space*, was compiled separately using the provided item data). The second set, which consists of less than 5% of the data, was used for testing and optimization, allowing us to refine our technique and tune its parameters. And the third set, which includes about 10% of the data, was used in the experiments described and analyzed in this report.

5 The Technique vs The Experiment

An important distinction should be made between the *SRR* technique and the experiment described in this report. Both the technique and the experiment leverage the provided data—however, the experiment is a simulation and a limited one at that. A key limitation is that the provided query data is confined to what the user was actually shown. That is, the search engine may have identified several pages worth of results in response to a *user-query*, but our dataset consists only of those pages actually seen by the user. Meanwhile, the concept behind the *SRR* technique calls for a search engine to deliver to the algorithm the top N items in response to a *user-query independent of the number of items ultimately shown to the user*. As a consequence, it is difficult, if not impossible, to simulate our technique using shown query results that are truncated because a user only viewed one or two pages. Even more generally, the use of historic data to demonstrate the consequences of a online ranking algorithm is intrinsically limited by the fact that one cannot be certain how users would have behaved if presented with different results. Nonetheless, we have done our best to conduct the most fair and informative experiment and analysis.

6 The Experiment

A key feature of *SRR* is that it can only reorder the results of a query if a user has previously clicked on an item during an online session. Consequently, *SRR* can only affect the subset of queries that occur in a session with previous clicks. We denote this subset of queries as ζ and limit our experiment and analysis to this set. It turns out that 25% of all queries are in ζ. Moreover, 28% of all clicks and 36% of all purchases occur within the query resultsets of ζ, since there is a correlation between previous clicks and clicks/purchases in a query. Thus an impact to this subset can have a significant impact overall.

The goal for the experiment is to simulate *SRR* using the provided historical query data, which is limited to what users were actually shown. An online implementation of our technique would receive the top N items from the search engine and reorder them prior to showing any results to a user. Because of this, the final ordering would be independent of the total number of items actually seen by a user (determined by the number of pages a user clicked through). Our test set χ therefore consists solely of queries within ζ where either all items in the query resultset or at least $N = 100$ items were shown to the user. For example, if a user stops searching after viewing only the 16 items on the first page (the default number of items on the first page), this query is not included in χ, since it is unknown which other items would have been considered for reordering. On the other hand, if the search engine found only 13 items in response to a query, we have the complete query resultset and can therefore determine how *SRR* would have reordered the shown items. Similarly, if more than $N = 100$ were shown to the user, we can determine the reordering regardless of whether the query resultset is truncated since *SRR* only considers and reorders the first $N = 100$ items. χ makes up 30% of ζ, accounting for a total of 7.5% of all queries.

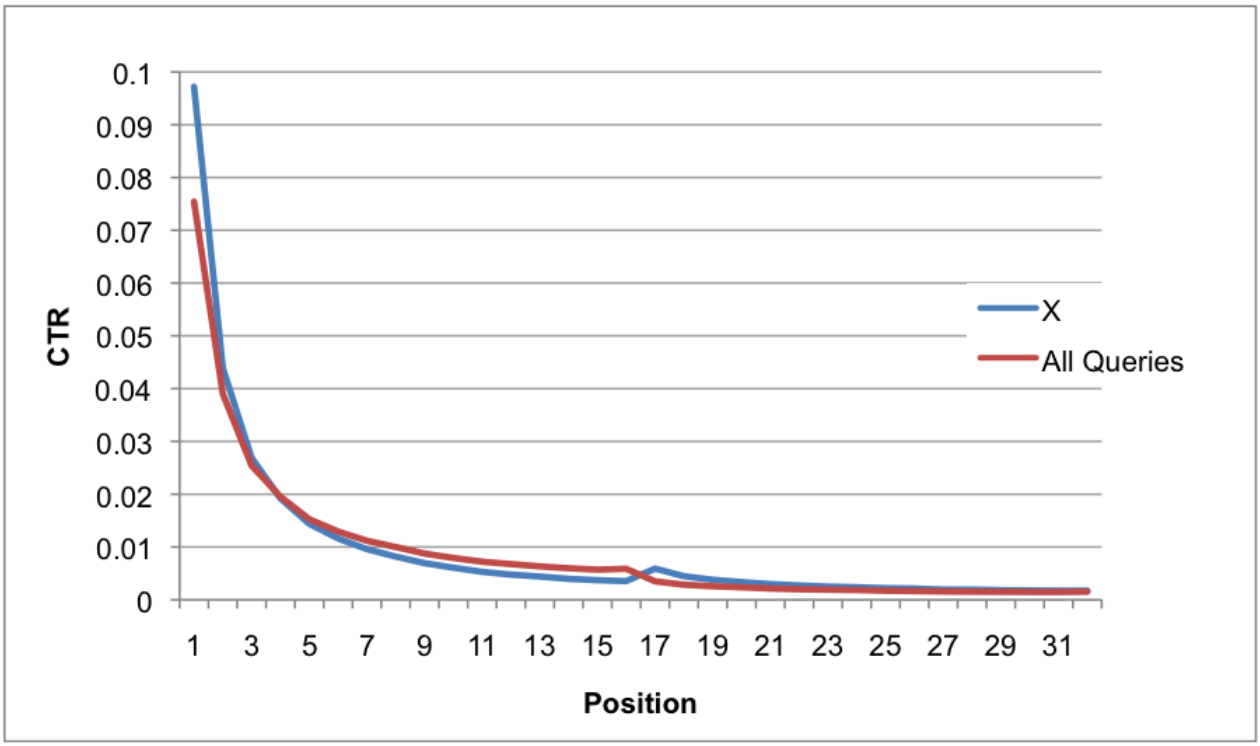

Figure 1: CTR as a function of position from the original data (i.e. not reordered) for the test set χ (blue) and all queries (red). *SRR* outperforms the original ordering in all metrics: $\Delta\ \mathcal{C} = 16.9\%$, $\Delta\ \mathcal{P} = 8.8\%$, and $\Delta\ \mathcal{C} = 7.9\%$

To construct χ we must discard all queries with a number of shown results less than $N = 100$ that are also divisible by 16. The reason for this is that Walmart.com provides two options for the number of items shown per page: 16 or 32. Thus, by performing the experiment on this subset of the data we precluded queries where the top N items are not available to our algorithm. The choice of $N = 100$, meanwhile, is somewhat arbitrary and was made by balancing our desire for a large test set with our desire to use a value appropriately large for an online implementation. It is therefore quite possible that a larger value of N (e.g. 1000) would achieve better results in the actual online scenario.

While we must constrain χ in this way due to the nature of the available data, we stress that this subset is certainly biased with respect to queries in general. For starters, queries with short resultsets are more likely to have all of their resultset seen by a user, and therefore, are more likely to be included in χ. It is not clear, however, if this particular bias tends to under- or overestimate the effectiveness of *SRR* since, as we will show, *SRR* is more effective on longer query results. Similarly, highly qualified queries—e.g. through the use of category or price filters—tend to have shorter resultsets, and hence, are more likely to be included in χ.

Just as interesting are the ways in which the queries and resultsets of χ are not biased. In Fig. 1 we show click-through-rate (CTR) as a function of position of the original data (i.e. not reordered) for both χ and

query results in general. The two curves shown in the figure are quite similar, indicating that the quantity and distribution of clicks within χ are essentially representative of those in general. A few other features of this figure warrant brief comment. First, we see that χ has a larger CTR for the top two positions. This is likely due to the fact that highly qualified queries, which have shorter results and higher CTRs, make up a higher proportion of χ than queries in general. While this difference does results in a slightly higher frequency of clicks in χ, we have no reason to believe that this alone results in a significant bias. Next, we see in the red curve a discontinuity appears at position 17 as a result of the pagebreak. In the entire dataset, the majoriy of queries are truncated at the first page, leading to significantly fewer views of items on other pages and thus fewer clicks. This discontinuity is absent from the blue curve since χ has a less severe dropoff in viewership from one page to the next. The bump at position 17 can be ascribe to users' tendency to disproportionately click on the topmost item shown on a page. Indeed, if we normalize for number of pages viewed by a user, we see this bump at 17 in the overal dataset as well. Consistent with this analysis, we find another smaller drop in the red curve and smaller bump in the blue at position 33.

7 Metrics

A true test of the effectiveness of *SRR* would require online A/B testing. In the meantime, we can simulate the effect of *SSR* by running it on historical data and examining the new positions of clicked and purchased items. Assuming the user would have clicked or purchased the same items in this new ordering, we can compare the distribution of clicks in the original ordering to that yielded by *SRR*.

To compare two orderings of shown items, we focus on three key metrics. Our primary metric is the first-page CTR $\mathcal{C}$, defined as the likelihood that an item presented on the first page receives a click. Special attention to the first page is reasonable, since 77% of all queries are only one page long. This shows the importance of bringing desirable items to the first page and it motivates our focus on $\mathcal{C}$. We calculate this metric by counting the number of items in first-page positions that were clicked and dividing by the number of total items in first-page positions. More formally

$$\mathcal{C} = \frac{\sum_{q \in Q} \sum_{i=1}^{L_q} \mathbb{1}\left\{click \ @ \ i \wedge i \leq 16\right\}}{\sum_{q \in Q} \sum_{i=1}^{L_q} \mathbb{1}\left\{i \leq 16\right\}}, \tag{3}$$

where Q is a set of query results, L_q is the number of results for query q, and the number 16 is due to the fact that 16 items are shown on a page.

Our second metric is the purchasing rate of items on the first page $\mathcal{P}$. This is similar to $\mathcal{C}$, except here we consider purchases per first-page item instead of clicks. It is calculated as the number of items in first-page positions that were purchased divided by the total number of items in first-page positions. The importance of position for purchases is even stronger than that for clicks. While 76% of all clicks were presented on the first page, that number is 88% for purchases. Formally, we have

$$\mathcal{P} = \frac{\sum_{q \in Q} \sum_{i=1}^{L_q} \mathbb{1}\left\{purchase \ @ \ i \wedge i \leq 16\right\}}{\sum_{q \in Q} \sum_{i=1}^{L_q} \mathbb{1}\left\{i \leq 16\right\}}. \tag{4}$$

The first two metrics focus on whether or not a desirable item was presented on the first page. To obtain a more granular picture of where desirable items are positioned, we also compute a third metric which we call *click-position score* $\mathcal{S}$. Somewhat similar to normalized discounted cumulative gain (NDCG), which is a common metric of search engine results, this score weighs the value of a clicked item by its position, giving higher weights to items closer to the top. For $\mathcal{S}$, the weight given to a click in position i is the CTR at position i Γ_i. In this way, we equate how often users click on a certain position to how valuable it is to put a desirable item there. Formally, we define *click-position score* as

$$\mathcal{S} = \frac{1}{|Q|} \sum_{q \in Q} \sum_{i=1}^{L_q} \mathbb{1}\left\{click \ @ \ i\right\} \Gamma_i. \tag{5}$$

These metrics give a sense of an ordering scheme's success. Each one looks at a slightly different aspect of the ordering. Indeed, optimizing for one metric does not necessarily optimize for the others. We focus our optimizations and primary analysis on $\mathcal{C}$ as we believe it is the simplest and has the clearest impact to overall CTRs.

8 Results

Figure 2 shows the clicks-position score $\mathcal{S}$ as a function of each of the coefficients C_s discussed in sections 1 and 2. Here, we vary a single coefficient, corresponding to one of the five similarity spaces—*click-space*, *cart-space*, *query-space*, *title-space*, or *item-space*—while setting the others to zero. The ranking score for each of the top $N = 100$ items is then determined by only two terms, as per Eqs. 1 and 2, as

$$\sigma(A) = \sum_{B \in P} C_s \cdot (J_s(A, B))^{\alpha_s} + \Gamma_i, \tag{6}$$

where, as before, P is the set of previously clicked items and Γ_i is the CTR of the original position of item A. (Note that the exponents α_s are held fixed during these measurements.) Thus, since Γ_i is a monotonically decreasing function of position i, when $C_s = 0$, *SRR* returns the original ordering.

In each case, as C_s is increased from zero, the degree of reordering is enhanced. And for each coefficient, this reordering is accompanied by an increase in $\mathcal{S}$ indicating a greater concentration of clicked items, on average, in the top positions of query results as compared to the original ordering. Moreover, with the exception of *title-space*, the score increases monotonically with each coefficient. In the case of *title-space*, a broad maximum can be seen around $C_s = 0.05$. This indicates that an optimal combination between the contributions from *title-space* and CTRs exists, which maximizes $\mathcal{S}$. For all other coefficients, however, a maximum cannot be found. Rather, the value of $\mathcal{S}$ asymptotically increases with value of C_s, which demonstrates that the optimal average ordering, according to the metric $\mathcal{S}$, is determined solely by the similarity space irrespective of the original ordering.

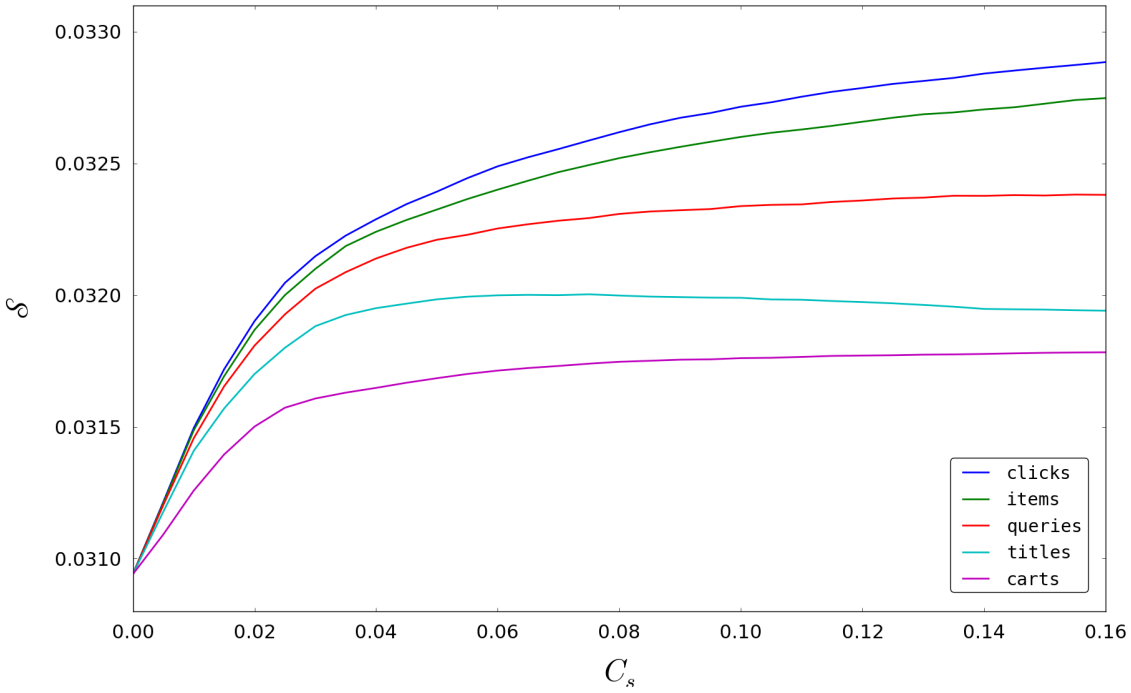

Figure 2: Clicks-position score $\mathcal{S}$ as a function of individual similarity space coefficients C_s. For each curve, a single coefficient is varied with all other set to zero.

We employ hueristics to manually optimize the coefficients C_s and exponents α_s, which are outside the scope of this report. We do want to emphasize two important points regarding this optimization, however. First, as previously mentioned, we conduct this optimization on a set of data that is both disjoint with that used to construct the similarity spaces indexes and with that used to conduct the experiment whose results are shared below. Second, we find that the optimal configuration includes nonzero values for all five coefficients showing that their contributions are not fully redundant.

With our optimized combination of coefficients we find that *SRR* significantly outperforms the original ordering on queries in χ. As shown in Fig. 3, the reordering achieves a 16.9% increase in $\mathcal{C}$ compared to

the original ordering. The implication is that items on the first page of the reordering are 16.9% more likely to be clicked than those in the original ordering. With over 2M queries in our test set $\mathcal{X}$, these results are statistically significant, producing a 95% confidence interval of [16.4%, 17.4%] for $\mathcal{C}$.

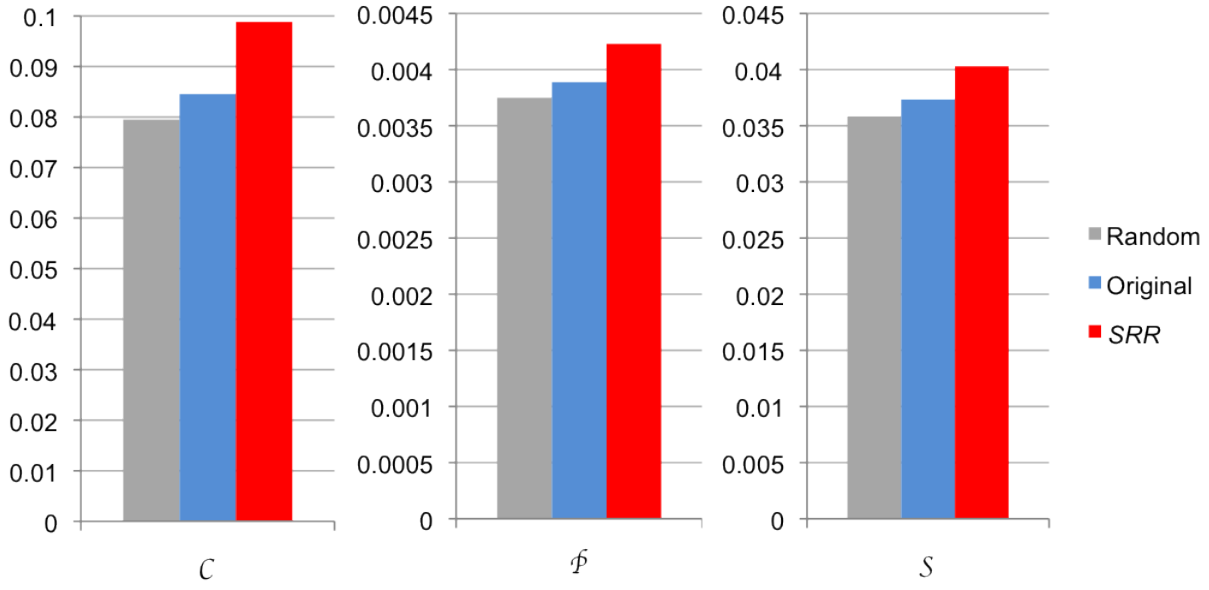

Figure 3: Comparison of the original ordering (blue) to that of random reordering (grey) and *SRR* (blue) using all three metrics: front-page CTR $\mathcal{C}$, front-page purchase rate $\mathcal{P}$, and click-position score $\mathcal{S}$.

Notably, we see these results despite the benefit the original ordering receives from "position bias", the well-established phenomenon that users tend to click on items presented at the top of a list. Figure 1 shows how dramatically more likely Walmart.com users are to click on items positioned near the top. It seems reasonable to suppose that "position bias" contributes to this distribution of CTRs. We compensate for the large CTR of the first two positions by setting the insert position $I_0 = 2$ such that the top two items remained fixed. While this does improve the metrics for *SRR*, we emphasize that choosing $I_0 = 0$ would not qualitatively change our results. Another obstacle overcome by *SRR* is that there is little room for improvement in first-page CTR, since the majority of clicks in $\mathcal{X}$ (58%) are already on the first page.

To ensure that these results are not due to an underlying structure of the data, we constructed a Random Re-Ranker *RRR*. *RRR* operates exactly as *SRR* does, except instead of calculating various similarity scores, it uniformly chooses at random a number between 0 and 1. This randomized algorithm significantly hurts the results in all three metrics, suggesting that *SRR* achieves its success by intelligently deciding which items are more likely to be clicked.

Indeed, Table 4 compares the average CTR of items *SRR* moves on or off the first page to those chosen randomly. By accurately determining a users preference for certain items, *SRR* promotes items with a significantly higher CTR than it would by blindly picking items from other pages. Similarly, it demotes first-page items with a lower CTR than an average item on the first page. With these successful decisions, *SRR* manages to outperform the original ordering in the face of a number of disadvantages.

	CTR of promoted items	**CTR of demoted items**
SRR	6.24%	1.79%
RRR	2.42%	3.67%

Table 4: CTR of items promoted to the first page and demoted off the first page. *SRR* siginificantly outperforms random reordering in both categories.

Taking a closer look, we see that the performance of *SRR* is correlated to query length L_q. Figure 4

shows the percent increase in $\mathcal{S}$ as a function of L_q. While it on average outperforms the original ordering for query results of any length, it gains greater improvement as L_q increases. This is an impressive feature of *SRR*, since as L_q increases, the density of clicked items decreases making them more difficult to select. At the same time, longer results provide greater opportunity for improved ordering, which *SRR* successfully exploits as shown by the data.

The specific trends within the data are more difficult to understand. There appear to be linear trends for $L_q \in [1, 16]$ and $L_q \in [17, 32]$, which correspond to single and double page results, respectively. Meanwhile, the trend for $L_q > 32$, while generally increasing, is less clear. We speculate that these trends are due to idiosyncrasies in user behavior but details remain unclear to us.

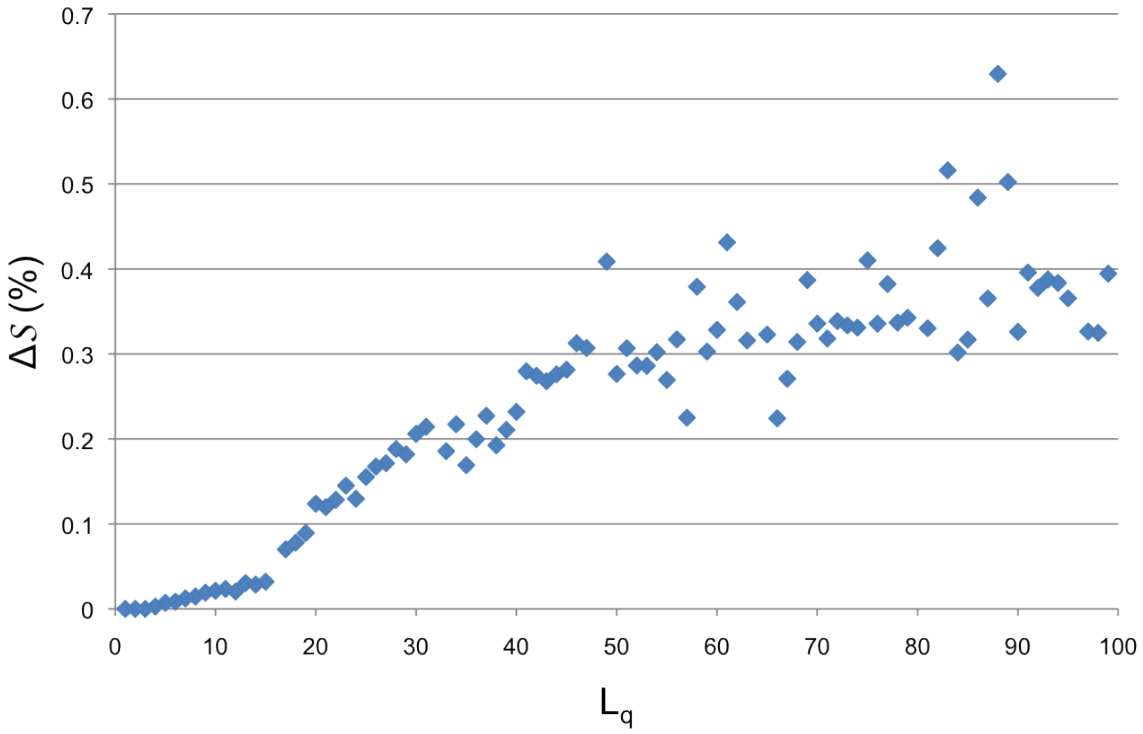

Figure 4: Change in click-position score $\mathcal{S}$ as a function of query length L_q.

9 Discussion

We have established that *SRR* significantly improves the ordering of search results by all three metrics on the dataset χ. Note that χ represents 30% of ζ (the set of all queries occuring in sessions with previously clicked items) while the remaining 70% of query results were truncated. We call this set of truncated query results τ. As discussed in Section 6, we cannot accurately test *SRR* on τ since it would have performed differently on this set of queries. [1]

Extrapolating from χ to τ is difficult due to inherently different characteristics of the two sets and doing so potentially introduces significant error. For instance, query results in χ were either completely shown or had at least the first $N = 100$ items shown to the user. This may result in disproportionately large numbers of cases where users were dissatisfied with items in the top positions. (Although, the CTR versus position data of Fig. 1 suggests this is not the case.) Perhaps more significant, queries in χ are on average shorter than those in τ and, as demonstrated by the data in Fig. 4, *SRR* performs better on longer queries.

[1] We ran *SRR* on all of ζ, treating truncated query results as complete ones of the length shown to the user. The results were similar to those of χ: $\Delta\ \mathcal{C} = 12.5\%$, $\Delta\ \mathcal{P} = 8.7\%$, and $\Delta\ \mathcal{S} = 8.2\%$. We omitted these results from the main paper because we feel that testing on all queries in ζ is not a representative experiment.

While it is unclear whether *SRR* would do better or worse on τ, we can infer a very loose lower bound on $\mathcal{C}$ over all of ζ by assuming that *SRR* performs as badly as it reasonably could on τ. When run on χ, *SRR* on average demotes 6.8% of all first-page clicks, but replaces them with even more clicks from other pages. In the worst case, we assume that when run on τ, it would demote 6.8% of first-page clicks but replace them with 0 clicks from other pages. Assuming this extremely pessimistic lower bound for the 70% of queries in ζ that make up τ, *SRR* would still sligthly outperform the original ordering across all queries (+0.004%).

If instead we assume that *SRR* would perform similarly on τ as it does on χ, we can get a more plausible estimate of the impact that the algorithm would have overall. Broadening the scope of our results to all queries (even those outside of ζ which *SRR* could not impact), we get an overall increase in $\mathcal{C}$ of 4.5% and an overall increase in $\mathcal{P}$ by 3.0%. To get a better sense of what this impact means, consider the set of purchases that occur on the first page, which comprise 88% of all purchases. This implies an overall increase of 2.6% in purchases, which is quite substantial given the scale of Walmart.com.

10 Conclusion and Future Work

We have reported on a novel technique of leveraging existing session information in order to improve the ranking of product search results. The approach taken by *Session Re-Rank* is to compare items previously clicked during a user session to the top N items returned by an existing search engine and to assign and rank by scores based upon their similarity. *SRR* utilizes historical data of queries, their results, and user actions to construct several indexes, each representing similarity space in which items can be compared via the Jaccard similarity of sets of objects. The similarity spaces we have used (*click-space*, *cart-space*, *query-space*, *title-space*, *item-space*) are not exhaustive—certainly others can be realized. We have provided compelling evidence using these 5 spaces, however, that *SRR* can significantly improve the quality of Walmart.com's existing query result rankings. Based on the statistically signicant improvements observed using first-page CTR $\mathcal{C}$, first-page purchase rate $\mathcal{P}$, and *click-position score* $\mathcal{S}$, we believe *SRR* would be a good candidate for A/B testing.

There are more experiments and ideas we would like to explore. Only five distinct vector spaces have been implemented to measure similarity between items based on historical session data. There certainly are more possibilities. A couple of unexplored ideas for vector spaces are considering items similar if they were purchased together in the same user session or similar if they are "close" in the tree of shopping categories. We did not experiment with the similarity metric used to compare items in each vector space. For this project, we chose to use Jaccard similarity for every vector space. Using only cosine similarity, or a mix of both Jaccard and cosine, would be worth evaluating. Another place for improvement is the clustering of queries in the *query-space* index.