Effective and Efficient User Interaction for Long Queries

Giridhar Kumaran*
Microsoft Live Labs
One Microsoft Way
Redmond, WA 98052, USA
giridhar@microsoft.com

James Allan
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts Amherst
140 Governors Drive, Amherst, MA 01003, USA
allan@cs.umass.edu

ABSTRACT

Handling long queries can involve either pruning the query to retain only the important terms (reduction), or expanding the query to include related concepts (expansion). While automatic techniques to do so exist, roughly 25% performance improvements in terms of MAP have been realized in past work through interactive variants. We show that selectively reducing or expanding a query leads to an average improvement of 51% in MAP over the baseline for standard TREC test collections. We demonstrate how user interaction can be used to achieve this improvement. Most interaction techniques present users with a fixed number of options for all queries. We achieve improvements by interacting *less* with the user, i.e., we present techniques to identify the optimal number of options to present to users, resulting in an interface with an average of 70% fewer options to consider. Previous algorithms supporting interactive reduction and expansion are exponential in nature. To extend their utility to operational environments, we present techniques to make the complexity of the algorithms polynomial. We finally present an analysis of long queries that continue to exhibit poor performance in spite of our new techniques.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Query formulation

General Terms

Algorithms, Experimentation, Performance

Keywords

User Interaction, Interactive Retrieval Efficiency, Query Reduction, Query Expansion, Query Analysis

*This work was done while the author was a graduate student at University of Massachusetts Amherst

SIGIR'08, July 20–24, 2008, Singapore.

1. INTRODUCTION

Past work suggests that richer expressions of information need by users can be leveraged to improve search performance. The richer expression can take the form of longer than usual queries, i.e., more than two to four terms in length [17], inclusion of additional terms the user believes are related to the query [14], identifying documents containing similar information [23], identifying topics the query is related to [16] and so on. We refer to all such expressions as *long queries*. Handling long queries is however difficult as they usually contain a lot of noise. This noise is in the form of extraneous terms that the user believes are important to conveying the information need, but in fact are confusing to automatic systems. Creating a more concise query by identifying and retaining the important terms in the long query (query *reduction*) is thus an important and challenging problem that needs to be solved. As opposed to automatic means, interactive query reduction (IQR) [17] is particularly effective (Section 2) in solving this problem.

Automatic query *expansion* techniques like pseudo-relevance feedback (PRF) [19] also help improve performance for certain types of long queries. Greater gains can be obtained from interactive query expansion (IQE)(Section 2), which involves asking the user to help remove wrong terms suggested by the automatic technique.

User interaction requires cognitive and physical effort from the user. Guided by the philosophy that users must get the maximum benefit (effectiveness) for their investment of time and effort, we explore ways to make user interaction for long queries more effective in Section 3. Determining when to interactively reduce or expand a query (Figure 1) can deliver the sort of improved effectiveness (over 50%) we seek. Automatic techniques to consistently determine whether to expand a query or not have had limited success [7]; we will show we can perform selective interactive reduction and expansion (SIRE) using implicit feedback from the user.

Our past explorations of IQR and IQE techniques [17] involved asking a user to select from ten options for each and every query. This can be detrimental to the user experience. Developing techniques to identify and present a minimal set of options to users is thus important. After demonstrating the similarity of the problem with *Set Cover*, a NP-Complete optimization problem, we utilize a greedy algorithm to provide an approximate solution (Section 4). Additionally, we exploit the presence of redundant information in the interface to further prune the set of options presented to the user.

P@5	P@10	NDCG@15	MAP	What is the effect of Turkish river control projects on Iraqi water resources?
0.600	0.400	0.619	0.706	▼ *turkey river iraq resource water*
0.000	0.000	0.000	0.020	▽ turkey iraq
0.000	0.000	0.000	0.004	▽ river water
0.600	0.400	0.551	0.676	▽ turkey river iraq water
0.000	0.000	0.000	0.008	▽ river project water
0.600	0.300	0.598	0.698	▽ turkey river iraq project water
0.000	0.000	0.000	0.008	▽ river resource water
0.000	0.100	0.041	0.050	▽ turkey iraq control
0.400	0.200	0.366	0.286	▽ turkey river iraq
0.600	0.300	0.417	0.530	▽ turkey iraq water

Turkey's South East Anatolian Project (4): Water supply a thorny issue - Downstream neighbours resent Turkey's control... channel under the embankment. On that occasion, marked by the presence of President Turgut Ozal, **Turkish** engineers staunched the Euphrates for a period of 30 days, trapping the **water** behind the dam for future irrigation and power generation. Syria and **Iraq, Turkey's** downstream neighbours, both protested.

Table 1: The top ten sub-queries presented for the *description* portion of TREC Topic 610. The original query is provided in the header row. Using a tabbed interface, users could select a sub-query and view the associated snippet in a window to the right. In our example, a hypothetical user has selected *turkey river iraq resource water*, and is shown the associated snippet of text (which turns out to be relevant). Clicking on other sub-queries will result in different snippets being loaded in the display area. The corresponding evaluation measures for each option are included in the left hand portion of the table (not part of the interface). In this example three out of ten options had a MAP better than the baseline of 0.583.

Our previous work on IQR and IQE [17] involved analyzing all possible combinations of the terms in the long query (**sub-queries**) or set of terms suggested by PRF (**expansion sets**) to determine the set of top *options*[1] to present to users. Such a technique is difficult to realize in practice due to the exponential number of options that need to be analyzed. In Section 5 we present a technique based on analyzing the properties of *ideal* queries, and using those observations to prune the option search space.

While our techniques helped improve performance of a significantly larger fraction of long queries compared to automatic techniques, there still remained a few queries that were not amenable to either automatic or interactive handling. In Section 7 we analyze such queries and categorize the reasons for their failure. We believe this analysis will be useful not only to determine the categories of problems that have been addressed by our techniques, but also to help plan strategies to tackle those that were not.

2. MOTIVATION

2.1 Past approach

In this section we provide an overview of the interaction technique for long queries that we build on in this paper. The right side of Table 1 provides an example of the interface provided to users in response to a long query for IQR. The long query in the example is the description portion of TREC Topic 601. Sub-queries are presented to the user along with a corresponding top-ranking snippet of text retrieved by each of them. Which sub-queries to select for display from the available exponential number of choices is based on a technique described in the next paragraph. The tabbed interface allows the user to click on each sub-query, view the associated snippet, and select the most promising one as their new query. The table also contains various performance measures (not shown to the user) that provide an idea of the utility of each sub-query. Notice that simple addition and deletion of terms can produce marked changes in performance. The interface for IQE is similar: the sub-queries are replaced by subsets of terms (expansion sets) from a set identified using PRF.

To identify top-ranking options we represented each of the 2^n options as a graph constructed with the constituent terms as vertices, and the mutual information [4] between the terms as edge weights. The maximum spanning tree [5] was identified on each graph, and its weight used to represent the quality of the option. After ranking the entire set of options by the weight of their corresponding maximum spanning trees, the top ten were selected.

In previous work [18] we performed user studies that demonstrated that users could use such an interface to select better alternatives, and obtain significant improvements in performance. This performance was also better than that acheived by simply using the title portion of the TREC query. Since we are improving on that interaction technique, in this paper we will confine ourselves to performing simulated user studies. We hypothesize that improvements such as more efficient background processes, fewer options presented to users, and better quality of options will naturally extend to improving the interaction experience and performance.

Table 2 shows the best performance that can be achieved under various conditions. All results are reported for 249 TREC *description* queries from the Robust 2004 track. We will treat the description portion of TREC queries as long queries for our experiments. Baseline refers to a query-likelihood (QL) run using the Indri search engine [24], while PRF refers to automatic query expansion using PRF[2]. "Upper Bound" refers to the situation when the best sub-query and best expansion set was used for query reduction and expansion respectively. In other words, if we had access to an oracle that always provided us the best sub-query and best expansion set for a query, we can obtain the indicated upper bound on performance. "Interaction Upper Bound" refers to the upper bound on the performance that can be obtained from user interaction, i.e. the user always selects the best option from the ten presented.

[1]We refer to sub-queries and expansion sets collectively as options.

[2]The performance metrics reported for PRF are upper bounds. We performed comprehensive parameter sweeps to determine the best parameter settings. The performance in practice will be lower as sub-optimal parameters will be learned for each collection by training on other collections.

System	P@5	P@10	NDCG@15	MAP
Baseline (QL)	0.472	0.397	0.379	0.240
PRF (Best)	*0.514*	*0.442*	*0.423*	*0.288*
Query Reduction				
Upper Bound (UB)	**0.799**	**0.671**	0.626	**0.366**
Interaction UB	**0.634**	**0.528**	0.498	*0.300*
Query Expansion				
Upper Bound (UB)	**0.738**	**0.643**	0.587	**0.368**
Interaction UB	**0.571**	**0.480**	0.447	*0.292*

Table 2: The utility of IQR and IQE. Italicized values indicate that the scores are significantly better than the baseline, while those in bold are significantly better than PRF. Statistical significance was measured using a paired t-test, with α set to 0.05.

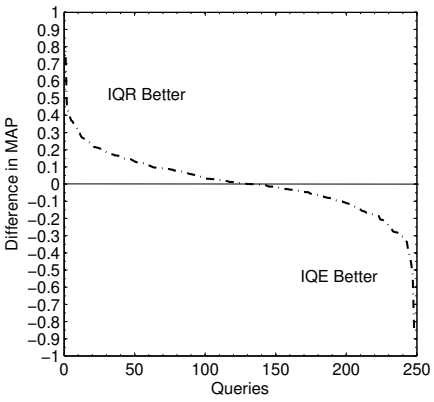


Figure 1: Difference in MAP due to Selective IQR and IQE.

2.2 Opportunities

The results in Table 2 for query reduction and expansion show that user interaction can lead to significant improvements in performance for long queries. Further improvements can be obtained if we selectively invoke IQR or IQE. Figure 1 shows the ordered distribution of the difference between the potential gains due to IQR and IQE. Some queries are better suited for IQR, while others can be better improved through IQE. If we can selectively invoke IQR or IQE for each query we can potentially obtain a 51% (from 0.240 to 0.363, compared to 0.300 and 0.292 for only IQR and only IQE respectively) improvement in MAP over the baseline. Determining when to reduce and when to expand is similar in flavor to the problems of determining when to perform PRF [7] or when to perform stemming [9]: correct answers to either can lead to significant improvements in performance. The tremendous scope for improvement makes the reduce/expand problem worthy of further investigation. We will show in Section 3 that we can address this problem through implicit feedback from the user.

The current interaction paradigm involves always presenting users with ten options for all queries. There is clearly scope for reducing the number of options presented to users, especially when on average only three out of ten of them are better than the baseline. Figure 2 is a histogram of the number of options better than the baseline for each of the 249 queries we used for training. Clearly, a large fraction

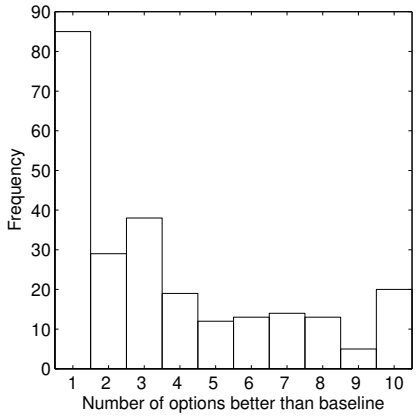


Figure 2: Distribution of the number of options in the ten presented to users that are better than the baseline query, for a set of 249 training queries.

of the options presented to users have no utility, and can potentially degrade the user experience. In Section 5 we present techniques that enable us to reduce the number of options we present to users significantly, without degrading performance.

IQR and IQE as reported in past work require a large amount of background processing. For query reduction the top ten options have to be selected from an exponential number of candidates since a query of length n has 2^n subqueries. Similarly, for query expansion, we need to analyze all 2^n combinations of expansion terms from the n suggested by PRF. Such exhaustive exploration of the sub-query space is infeasible in an operational environment. Also in Section 5 we will present a simple technique based on empirical observations that significantly reduces the search space, without sacrificing performance.

3. SELECTIVE INTERACTIVE REDUCTION AND EXPANSION (SIRE)

Tremendous gains in performance can be obtained by selectively expanding or reducing long queries. For each long query, our approach involved selecting the top five sub-queries and top five expansion sets and providing the user a merged list for interaction. The downside of this technique was that we risked losing potentially useful options ranked between six and ten. However, as Table 3 shows, this risk was insignificant when compared to the potential for improvement through SIRE. By viewing this mix of expansion and reduction options, along with a snippet of text to guide selection, the user can implicitly guide the system towards expansion or reduction of the query.

Table 3 summarizes the improvements in performance that can be achieved using the SIRE technique[3]. When IQR and IQE are used with five options, the performance is as detailed. However, when the options are combined (SIRE$_{comb}$),

[3]These results were obtained using an efficient processing technique described later in Section 5

System	P@5	P@10	NDCG@15	MAP	Avg. num. options
Baseline	0.472	0.397	0.379	0.240	-
IQR_5	0.554	0.469	0.448	0.274	4.9
IQE_5	0.555	0.466	0.435	0.292	5
$SIRE_{comb}$	0.65	0.552	0.521	0.347	9.9
IQR_{10}	0.634	0.528	0.498	0.300	9.7
IQE_{10}	0.571	0.480	0.450	0.292	10.0

Table 3: The subscript in the system name indicates the number of options presented to users. The $SIRE_{comb}$ technique involves merging IQR_5 and IQE_5. For comparison, performance of IQR and IQE with ten options is also provided.

the user can potentially achieve better performance than could be achieved with either IQR or IQE with not only five, but also ten options. If we started with using ten options each from IQR and IQE, we can expect even higher performance improvements. Another interesting aspect of the result is that MAP is significantly improved. IQR is primarily a precision enhancing technique, while IQE is both precision as well as recall enhancing. The advantages of each technique have thus been carried over to the hybrid SIRE technique in the form of improved MAP.

4. MINIMAL OPTION SETS

The technique to analyze the options makes use of co-occurrence information of the constituent terms. While this provides a good sense of the *cohesiveness* of the option, it does not inform the user of the relative utility of an option with respect to the other ones shown. Given that some options differ by just a single term, its quite likely that they might all direct the user to the same search space. In such cases it is wasteful to show similar options, and instead displaying a minimal subset of options that *covers* the original search space(s) might be better. This intuition forms the basis for our technique to prune the original set of options shown to the user. We introduce the *Set Cover* problem before going into the details of our technique.

The set covering problem [5] is a NP-complete optimization problem. An instance $(X, \mathcal{F})$ of the set covering problem consists of a finite set X and a family $\mathcal{F}$ of subsets of X, such that every element of X belongs to at least one subset in $\mathcal{F}$. Mathematically,

$$X = \bigcup_{S \in \mathcal{F}} S \tag{1}$$

The subset S is said to *cover* the elements in X. The goal is to find a minimum-size subset $\mathcal{C}$, $\mathcal{C} \subseteq \mathcal{F}$, whose members cover all of X i.e.

$$X = \bigcup_{S \in \mathcal{C}} S \tag{2}$$

Since finding the exact solution is NP-Complete, we used a greedy set cover algorithm [5] that works by selecting the subset that covers the most number of elements in X in an iterative fashion. The advantage of using the greedy algorithm is that it not only runs in time polynomial in $|X|$ and $|\mathcal{F}|$ but also returns a $(\ln(|X|) + 1)$ approximation to the solution.

4.1 Overlapping Search Results

We now show how the problem of finding minimal option sets can be cast as a set cover problem. Each option is used as a query to retrieve a set of ten documents. Let X be the union of sets of ten documents retrieved. The sets of ten documents correspond to the family $\mathcal{F}$ of subsets whose union is X. Our goal is to identify a minimal set of subsets $\mathcal{C}$ from $\mathcal{F}$ that cover X.

Table 4 shows the impact on performance metrics as well as the number of options presented to users due to the various techniques. We can observe in the case of "Set-Cover-based Pruning" for IQR that for an average decrease of two options per query, there is no (statistically) significant drop in performance. In the case of IQE, it is drastic: an average reduction of six options per query without significant performance loss. This result for IQE can be understood by considering the fact that query expansion results in a much longer query than the original, and the subtle differences between options (usually by a term or two) do not lead to radically different sets of documents being retrieved. The results for SIRE show that the pruning strategy works for it too, and performance comparable to IQR is achievable by showing 50% fewer options. In summary, judging by the insignificant drops in performance, we successfully retained the useful options and removed the redundant ones.

4.2 Identical Snippets

The user is guided in making a decision on which option to select using a snippet of text. This snippet is extracted from the top-ranking document that is retrieved when the option is used as a query. Frequently, the snippets returned by different options are the same, making the task of selecting an option difficult. Retaining a single option from the set that retrieves the same snippet can further decrease the number of options presented to users. The results for "Snippet-based Pruning" in Table 4 show the impact of this pruning strategy. With the exception of P@5 for IQE, this strategy results in an average reduction of approximately one option without significantly impacting performance.

5. OPTION ANALYSIS

A good query is long enough to describe key concepts but also short enough to avoid containing unnecessary terms. To determine the appropriate length of sub-queries, we plotted the distribution of the query lengths of the best performing sub-query for each query in our training set. Figure 3 shows the distribution, and compares it with the distribution of the lengths of the original queries. We can observe that the best sub-queries are never more than ten terms in length, with most having six or fewer. This observation informed the decision to restrict analysis of sub-queries to those of length less than or equal to six. Table 5 shows the impact on "Interaction Upper Bound" performance due to this restriction. The restriction not only results in a reduced number of sub-queries analyzed but also maintains the potential for improvement through IQR.

In a similar fashion, we analyzed the size of the best expansion subset for our training queries (Figure 4). We can observe that the best expansion sets are frequently between eight to twelve terms in length. This observation again informed the decision to restrict analysis of expansion sets to those of length less than or equal to twelve. The section for IQE in Table 5 conveys that this restriction actually helped

System	P@5	P@10	NDCG@15	MAP	Avg. # options
Baseline (QL)	0.472	0.397	0.379	0.240	-
Interactive Query Reduction (IQR)					
Interaction Upper Bound	0.634	0.528	0.498	0.300	9.7
Set Cover-based Pruning	0.619	0.523	0.488	0.293	7.5
Snippet-based Pruning	0.610	0.510	0.483	0.290	6.7
Interactive Query Expansion (IQE)					
Interaction Upper Bound	0.571	0.480	0.435	0.292	10.0
Set Cover-based Pruning	0.560	0.479	0.437	0.294	3.8
Snippet-based Pruning	0.535	0.454	0.421	0.285	2.1
Selective Interactive Reduction and Expansion (SIRE)					
Interaction Upper Bound	0.654	0.552	0.521	0.347	9.9
Set Cover-based Pruning	0.643	0.544	0.512	0.340	6.7
Snippet-based Pruning	0.629	0.53	0.505	0.335	5.5

Table 4: Effect of option-pruning strategies on performance metrics, and number of options presented to users.

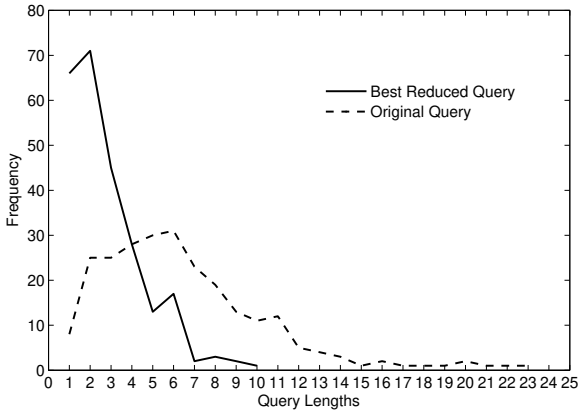


Figure 3: Distribution of lengths of original and best reduced queries

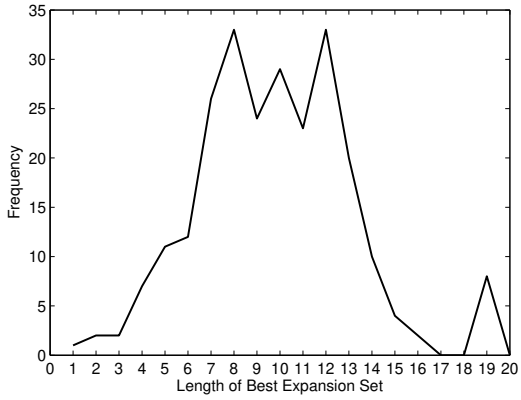


Figure 4: Length distribution of best expansion sets

System	P@5	P@10	NDCG@15	MAP
Interactive Query Reduction (IQR)				
Interaction UB	0.634	0.528	0.498	0.300
Interaction UB with Size Threshold	0.634	0.528	0.498	0.300
Interactive Query Expansion (IQE)				
Interaction UB	0.571	0.480	0.447	0.292
Interaction UB with Size Threshold	0.578	0.485	0.450	0.303

Table 5: Effect of thresholding the lengths of options analyzed. UB refers to Upper Bound.

avoid some bad options, and raised the potential for improvement through IQE.

6. EXPERIMENTAL SETUP

We used version 2.6 of the Indri search engine, developed as part of the Lemur[4] project. We used the query-likelihood variant of statistical language modeling as our baseline, and used the PRF mechanism based on relevance models [19] to generate terms for IQE.

As our data sets we used the TREC Robust 2004, Robust 2005 [26], TREC 5 ad-hoc [27] and HARD 2003 [2] document collections. The 2004 Robust collection contains around half a million documents from the Financial Times, the Federal Register, the LA Times, and FBIS. The Robust 2005 collection is the one-million document AQUAINT collection. The choice of Robust tracks was motivated by the fact that the associated queries were known to be difficult, and conventional IR techniques were known to fail for a number of them. The TREC 5 ad-hoc collection consists of TREC disks 1 and 2, and presented a standard ad-hoc retrieval setting. The HARD 2003 collection, a subset of the AQUAINT corpus and US government corpus containing 372,219 documents in all, was also selected since it was created for a track with focus on user interaction. The fifty queries in the Robust 05 data set overlap with those in the Robust 04 data set we used for training. However, since the collections are different, we do not stand the risk of over-fitting. The HARD data set uses the same collection as the Robust 04

[4] `http://www.lemurproject.org`

data set, but has a different set of fifty queries. Finally, the TREC 5 data set shares neither the queries nor the collection with the Robust 04 data set. We believe that this choice of test data sets will provide a comprehensive validation of our techniques. All collections were stemmed using the Krovetz stemmer provided as part of Indri. 249 queries from the TREC Robust 2004 track were used to study the impact of the various techniques presented in this paper, and to learn parameters used for thresholding. The remaining 150 queries, 50 each from the three remaining tracks, were used to test the generality of our techniques.

Al-Maskari et al. [1] have shown that measures based on cumulative gain [13] and precision correlate well with users' satisfaction of the results. For all systems, we report precision at five documents (P@5), precision at ten documents (P@10), normalized discounted cumulative gain at 15 documents (NDCG@15, as defined in [25]), and mean average precision (MAP).

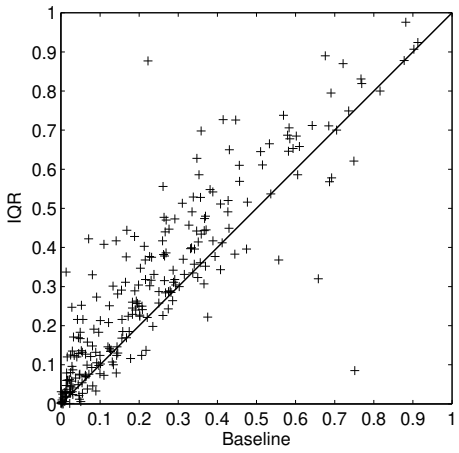
7. RESULTS AND ANALYSIS

In this section we analyze the effect of using our techniques on different data sets, the results of which are presented in Table 6.

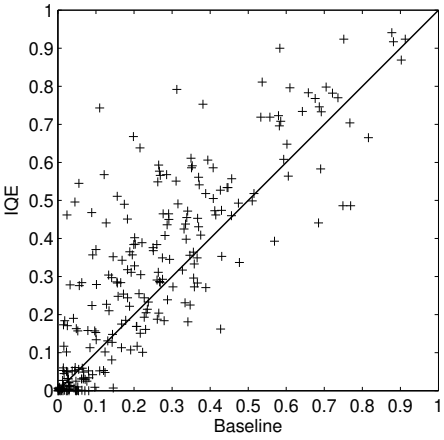
For all collections we notice trends similar to that observed for the Robust 04 data set namely the higher performance of the SIRE system compared to IQR and IQE with not only five options but also ten. SIRE remains competitive or better even after option pruning: with an average of six options it meets or beats ten-option IQR and IQE.

We now analyze the performance of IQR, IQE, and SIRE with respect to the baseline (automatic) system. Figure 5 shows the scatter plots of the MAP values of the baseline system with respect to each of the interactive techniques, for 249 Robust 04 queries. The line $y = x$ is included to identify the queries that were improved or hurt by each technique. A point above the $y = x$ line means that performance was improved through interaction, while a point below the line means that interactive retrieval did not help the query. We can observe that a larger fraction of queries was improved by IQR (Figure 5(a)) in comparison to IQE (Figure 5(b)). The plot for IQE (Figure 5(b)) has greater spread, and higher density in the upper left hand corner compared to IQR. This means that when IQE helps, it helps to a greater extent than IQR. However overall improvements are mitigated by the fact that IQE performs worse on already poorly performing queries. The SIRE system combines the best of IQR and IQE. Not only are there fewer queries below the $y = x$ line, but the density in the upper left hand corner is greater. These observations mean that SIRE provides a more comprehensive improvement over a set of queries.

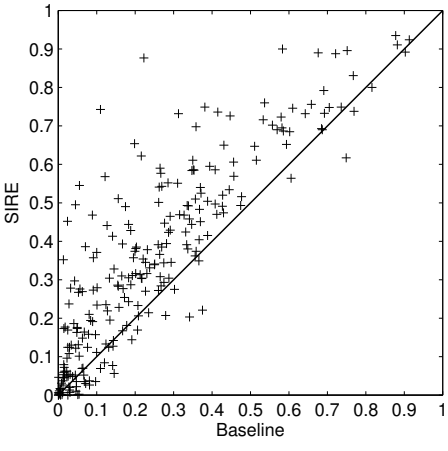
We now turn our attention to the lower left hand corner of Figure 5(c) - the area containing the set of queries that were not only poorly-performing to start with, but also were unaffected by IQR, IQE, and SIRE. We define poorly-performing queries as those that had a baseline, IQR, IQE, *and* SIRE MAP of less than or equal to 0.1. We analyzed each of the 45 such queries, and also the corresponding best reduced and expanded versions that were used to generate the "Upper Bound" scores in Table 2. For situations when the best reduced and expanded queries were themselves low-performing, it was clear that the user would have to enter a completely new query. Table 7 summarizes the failure categories we identified and suggests directions for future



(a) Baseline versus IQR



(b) Baseline versus IQE



(c) Baseline versus SIRE

Figure 5: Scatter plots of baseline performance (MAP) and performance due to IQR, IQE, and SIRE. IQR and IQE used ten options, while SIRE used a combination of five options from each of IQR and IQE

Corpus	System	P@5	P@10	NDCG@15	MAP	Avg. # options
Robust 05	Baseline	0.412	0.386	0.270	0.156	-
	IQR with 5 options	0.488	0.450	0.333	0.194	5.0
	IQE with 5 options	0.512	0.490	0.351	0.229	5.0
	SIRE with combined options	0.612	0.564	0.417	0.262	10.0
	SIRE with set cover-based pruning	0.604	0.556	0.415	0.257	7.2
	SIRE with snippet-based pruning	0.500	0.484	0.405	0.221	2.5
	IQR with 10 options	0.572	0.510	0.396	0.218	10.0
	IQE with 10 options	0.540	0.510	0.364	0.237	10.0
HARD 2003	Baseline	0.544	0.478	0.441	0.228	-
	IQR with 5 options	0.644	0.57	0.515	0.283	4.9
	IQE with 5 options	0.628	0.544	0.476	0.298	5.0
	SIRE with combined options	0.720	0.646	0.572	0.346	9.9
	SIRE with set cover-based pruning	0.720	0.636	0.570	0.343	6.8
	SIRE with snippet-based pruning	0.632	0.544	0.568	0.300	2.5
	IQR with 10 options	0.712	0.618	0.560	0.301	9.8
	IQE with 10 options	0.636	0.580	0.496	0.305	10.0
TREC 5	Baseline	0.384	0.322	0.305	0.163	-
	IQR with 5 options	0.372	0.308	0.327	0.154	4.9
	IQE with 5 options	0.352	0.302	0.312	0.166	5.0
	SIRE with combined options	0.444	0.370	0.389	0.202	9.9
	SIRE with set cover-based pruning	0.440	0.358	0.382	0.19	6.9
	SIRE with snippet-based pruning	0.348	0.298	0.378	0.161	2.3
	IQR with 10 options	0.468	0.374	0.386	0.171	9.7
	IQE with 10 options	0.368	0.312	0.320	0.168	10.0

Table 6: Summary of the results of using the SIRE and option-pruning techniques on test data sets.

Analysis	# queries
Term mismatch: new query required	24
System failure in identifying sub-query	
• Best sub-query incomprehensible to human	10
• Human could have identified it	3
• NLP techniques could have helped	4
System failure in identifying expansion set	4

Table 7: Breakdown of the analysis of low-performing queries. By NLP techniques, we refer to identification of phrases in the query and treating them as a unit.

work. "System failure in identifying sub-query (or expansion set)" refers to the situation when a better option was available, but the technique we used to rank the options failed to place it in the top 10. Of these, 3 of the options were of the type that a user with a similar information need could be expected to issue. Another 14 (10+4) of them would have been difficult for a human to come up with without a complete understanding of how the underlying search engine works. For the 4 queries for which NLP techniques would have worked, we expect that identifying noun phrases in the original query would have helped.

8. RELATED WORK

Irrespective of the environment, most user studies [17, 14] have reported improvements in performance from user interaction. Various studies [12, 15] also acknowledge the importance of good interfaces and decision support mechanisms to realize the potential of user interaction. Our work continues along this line and shows that given the right interactive technique and support mechanism, user interaction can provide great mileage.

An earlier exploration involving the user in IQE was carried out by Harman [8]. Positive user experiences were observed. Magennis and Van Rijsbergen [20] extended these investigations to simulated experiments on a larger scale. The idea of expanding the original query with sub-sets of predetermined length from the expansion term set is similar to ours, though the motivation was to find an upper bound on performance. Ruthven [21] extended this idea further by examining various query expansion techniques and performing user studies to compare IQR and IQE. His experiments showed that while there is potential for improvement through IQE, realizing the potential in practice is dependent on a number of limiting factors. Salton and Lesk [22] showed that user selection of expansion terms did not do as well as just having the system expand automatically. They reasoned that users did not know enough about how the IR system worked to do effective prediction. This problem can however be solved by showing users a preview of the information retrieved by each selection, as is the case in the interaction technique we developed (Table 1, [18]). We explored the idea of trying to find the appropriate query terms from a long description of a user's information need, and showed that automatic techniques supplemented by user interaction can deliver significant improvements in performance [17].

Numerous efforts have been made towards finding techniques for predicting query quality. Accurately predicting when a query would fail [7] can be used to attempt an alternate technique like PRF. Cronen-Townsend et al. [6] developed the clarity measure to serve as a predictive measure for tracking MAP. He and Ounis [11] explored a number of features derived from the query to determine query effectiveness. Recent work by Carmel et al. [3] attempted to

formalize the query difficulty problem. We have tackled a variant of the problem, namely determining if IQR or IQE is better suited for a query.

Harman and Buckley [10] conducted a workshop to identify the reasons why search engines fail. They analyzed the performance and outputs of multiple participating sites and identified ten reasons for failure of information retrieval systems on TREC description queries. While the categories we report are not identical to theirs, we plan on expanding our current limited analysis to determine the categories of problems they found that our techniques helped solve.

9. CONCLUSIONS

We have presented techniques to improve the effectiveness and efficiency of user interaction for information retrieval using long queries. The SIRE technique has been shown to be an extremely effective way to capitalize on the strengths of the IQR and IQE techniques. Presenting users with the right number of options is an often-ignored aspect of interactive information retrieval. We have developed a sound framework for identifying a minimal set of options, and demonstrated that this technique retains good options and removes the redundant ones. We hypothesize that this technique, which can be used in any interactive environment, will enhance effectiveness by reducing the cognitive load on users. The exponential-sized analysis of options has been shown to be unnecessary, and reduced to polynomial-sized analysis without degrading performance. The ease with which the analysis process can be parallelized (different machines can analyze options of different lengths and follow up with a merge) and reduction in the complexity can pave the way for live deployment of the effective interaction techniques we have presented in this paper.

Some directions for future work include parsing long queries using NLP techniques to support user interaction. While the option analysis procedure doesn't involve any querying of the index, the option-pruning procedure requires querying the index to obtain top-ranked documents. A better technique to approximate the top-ranking documents will make the process more efficient. IQR results in a concise version of the long query that is potentially better. In future work we plan to explore using the query identified thorough IQR as a starting point for either automatic or interactive query expansion. This two-stage interaction could potentially improve effectiveness even further.

Acknowledgments

We wish to thank the anonymous reviewers of this paper for their very helpful comments. This work was supported in part by the Center for Intelligent Information Retrieval and in part by the Defense Advanced Research Projects Agency (DARPA) under contract number HR0011-06-C-0023. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

10. REFERENCES

[1] A. Al-Maskari, M. Sanderson, and P. Clough. The relationship between IR effectiveness measures and user satisfaction. In *30th ACM SIGIR Proceedings*, pages 773–774, 2007.

[2] J. Allan. The HARD Track Overview in TREC 2003. High Accuracy Retrieval from Documents. In *TREC 12 Proceedings*, 2003.

[3] D. Carmel, E. Yom-Tov, A. Darlow, and D. Pelleg. What makes a query difficult? In *29th ACM SIGIR Proceedings*, pages 390–397, 2006.

[4] K. W. Church and P. Hanks. Word association norms, mutual information, and lexicography. In *27th ACL Proceedings*, pages 76–83, 1989.

[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, II Edition*. MIT Press, 2001.

[6] S. Cronen-Townsend, Y. Zhou, and W. B. Croft. Predicting query performance. In *25th ACM SIGIR Proceedings*, pages 299–306, 2002.

[7] S. Cronen-Townsend, Y. Zhou, and W. B. Croft. A framework for selective query expansion. In *Thirteenth ACM CIKM Conference Proceedings*, pages 236–237, 2004.

[8] D. Harman. Towards interactive query expansion. In *ACM SIGIR '98 Conference Proceedings*, pages 321–331, 1988.

[9] D. Harman. How effective is suffixing? *JASIS*, 42(1):7–15, 1991.

[10] D. Harman and C. Buckley. The NRRC reliable information access (RIA) workshop. In *27th ACM SIGIR Proceedings*, pages 528–529, 2004.

[11] B. He and I. Ounis. Inferring query performance using pre-retrieval predictors. In *The Eleventh Symposium on String Processing and Information Retrieval*, 2004.

[12] S. Henninger and N. J. Belkin. Interface issues and interaction strategies for information retrieval systems. In *CHI '96 Proceedings*, pages 352–353, 1996.

[13] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.

[14] D. Kelly, V. D. Dollu, and X. Fu. The loquacious user: a document-independent source of terms for query expansion. In *28th ACM SIGIR Proceedings*, pages 457–464, 2005.

[15] J. Koenemann and N. J. Belkin. A case for interaction: a study of interactive information retrieval behavior and effectiveness. In *SIGCHI Conference Proceedings*, pages 205–212, 1996.

[16] G. Kumaran and J. Allan. Simple questions to improve pseudo-relevance feedback results. In *29th ACM SIGIR Proceedings*, pages 661–662, 2006.

[17] G. Kumaran and J. Allan. A case for shorter queries, and helping users create them. In *HLT-EMNLP Conference Proceedings*, pages 220–227, Rochester, NY, 2007.

[18] G. Kumaran and J. Allan. Adapting information retrieval systems to user queries. *IP& M: Special Topic Issue on Adaptive Information Retrieval*, In press, 2008.

[19] V. Lavrenko and W. B. Croft. Relevance based language models. In *24th ACM SIGIR Proceedings*, pages 120–127, 2001.

[20] M. Magennis and C. J. van Rijsbergen. The potential and actual effectiveness of interactive query expansion. In *20th ACM SIGIR Proceedings*, pages 324–332, 1997.

[21] I. Ruthven. Re-examining the potential effectiveness of interactive query expansion. In *26th ACM SIGIR Proceedings*, pages 213–220, 2003.

[22] G. Salton and M. E. Lesk. Computer evaluation of indexing and text processing. *J. ACM*, 15(1):8–36, 1968.

[23] M. D. Smucker and J. Allan. Find-similar: similarity browsing as a search tool. In *29th ACM SIGIR Proceedings*, pages 461–468, 2006.

[24] T. Strohman, D. Metzler, H. Turtle, and W. B. Croft. Indri: A language model-based search engine for complex queries. In *International Conference on Intelligence Analysis*, 2005.

[25] S. Vassilvitskii and E. Brill. Using web-graph distance for relevance feedback in web search. In *29th ACM SIGIR Proceedings*, pages 147–153, 2006.

[26] E. M. Voorhees. The TREC 2005 robust track. *SIGIR Forum*, 40(1):41–48, 2006.

[27] E. M. Voorhees and D. Harman. Overview of the fifth text retrieval conference (TREC 5). In *TREC 5 Proceedings*, 1996.