

independent interest. We run three sets of experiments highlighting the use of categorical information for both queries and documents. These experiments differ in how the intent distributions and document relevance are collected. In all cases, our result diversification algorithm outperforms all three commercial search engines.

2. RELATED WORK

One of the early influential work on diversification is that of Maximal Marginal Relevance (MMR) presented by Carbonell and Goldstein in [5]. In their work, a tradeoff between novelty (a measure of diversity) and the relevance of search results is made explicit through the use of two similarity functions, one measuring the similarity among documents, and the other the similarity between document and query. A parameter controls the degree of tradeoff. As there is no categorization of either the document or the query, diversification is conducted through the choice of similarity functions.

Zhai *et al* [19] point out that it is in general insufficient to simply return a set of relevant results. The correlations among the results are also important. This work is later formalized in [20], where Zhai and Lafferty propose a risk minimization framework for information retrieval that allows a user to define an arbitrary loss function over the set of returned documents. This function determines the "unhappiness" of the user for a given set of results. In order to put the theory to practice, however, requires one to specify the loss function. In Chapter 7 of [18], Zhai discusses the problem of dependent topic retrieval, and proposes two loss functions that have lead to a selection of diverse results. These loss functions depend on certain language models rather than categorical information about the documents.

Bookstein [3] and Chen and Karger [6] both consider information retrieval in the context of ambiguous queries. The basic idea in these works is that documents should be selected sequentially according to the probability of the document being relevant *conditioned* on the documents that come before. Bookstein's method, however, is limited by the fact that it requires explicit user feedback for relevance after every document is selected. Chen and Karger work around this limitation, and propose an objective function that aims to find at least one relevant document for all users. In both cases, the topic of the query is considered only tacitly.

Radlinski, Kleinberg, and Joachims [13] propose a learning algorithm to compute an ordering of search results from a diverse set of orderings. By iterating through all documents in each of the positions while holding fixed the documents in the other positions, they attempt to learn a "best" ranking of documents using user clicks. Their approach naturally produces a diverse set of results, as user feedback through clicks will diminish the value of similar documents. However, the topics of the query is only considered implicitly.

Radlinski and Dumais [14] observe that the top k results of a query might not be diverse enough to contain representative documents corresponding to various interpretations of the query, thus limiting the usefulness of personalized client-side reranking of results. They study the question of generating *related* queries in order to yield a more diverse set of documents. Their work is complimentary to ours and can be used to enhance the set of input documents to our algorithm.

Compared to work described above, we take a different ap-

proach that makes use of a taxonomy for classifying queries and documents, and creates a diverse set of results in accordance to this taxonomy. Such taxonomy are readily available through projects such as the Open Directory Project (ODP). We now review some work that also make use of taxonomies.

Ziegler *et al* [21] study diversification in a recommendation setting. They propose an approach that returns lists of recommendations that better cater to users' full range of interests by selecting lists that have low inter-list similarity." They demonstrate experimentally that real users prefer more diversified results. Their classification of items are taxonomy-based. However, they do not consider the relative importance of the different categories, nor how good a particular recommendation is. We take into account both in our model.

Vee *et al* [15] study diversification in the context of structured database with applications to online shopping. In their work, items are represented as a set of features, and the objective is to select a set of items that are as diverse as possible according to a lexicographical ordering of features. They show that it is impossible in general to simply apply post-processing to retrieved results, and give two algorithms that directly take into account diversification. As the lexicographical preferences are known in advance and directly used in the problem formulation, it constitutes a form of explicit diversification. However, the notion of relevance is suppressed in their work, as the objective is to select items that are distinct from one another. This may be fine for structured domains, but less applicable to web search where documents differ greatly in their ability to satisfy user needs.

Clarke *et al* [7] study diversification in the context of answering questions. They focus on developing a framework of evaluation that takes into account both novelty and diversity. In their work, questions and answers are treated as sets of "information nuggets", and relevance is a function of the nuggets contained in the questions and the answers. The notion of nuggets can be mapped to the notion of categories in our paper. The main difference between their work and ours is that we take into account the relative importance of different nuggets (as distribution of categories), and the fact that different documents containing the same nugget may satisfy the user to different extent (as the relevance score of a document for a given category).

Finally, we point out that there has been a dearth of work on IR metrics that take into account both diversity and relevance. We believe this is an important area that deserves more research attention, and propose our own generalizations of classical IR metrics in Section 4. Review of related work will be presented then.

3. RESULT DIVERSIFICATION PROBLEM

We begin by explaining our assumptions and formalize an optimization objective for minimizing average user dissatisfaction. We then analyze the complexity of this problem, and propose a simple greedy algorithm that well approximates the objective in general and is provably optimal in the special case of single-category documents. For expository, proofs have been deferred to the Appendix.

3.1 Preliminaries

Assume there exists a taxonomy of information, and that user intents are modeled at the topical level of this taxon-

omy. Both queries and documents are categorized according to this taxonomy. Denote the set of categories to which a query q belongs as $C(q)$, and that for a document d as $C(d)$. Note that a query or a document may belong to multiple categories. For given query q and given document d , their categories need not overlap, i.e., $C(d) \setminus C(q)$ may be empty.

Further assume that there is a known distribution that specifies the probability of a given query belonging to given categories, $P(c|q)$, and that our knowledge is complete, i.e., $\sum_{c \in C(q)} P(c|q) = 1$. This distribution can be estimated in a number of ways in practice. In our implementation, we employ the query classification algorithm described in [1].

Let $V(d|q; c)$ denote the quality value of a document d for query q when the intended category is c , meant to capture the *relevance* of the document. Without loss of generality, let its value be in the range of $[0; 1]$. We give a *probabilistic interpretation* to the quality values: they approximate the *likelihood* of the document satisfying the user intent given the query. This value can be estimated using a variety of techniques. In our implementation, $V(d|q; c)$ is determined by a scoring function based on the content of the document and the query, weighted by the likelihood that the document belongs to a particular category.

We make a certain *independence* assumption: given a query and the category of the intent, the conditional probabilities of the two documents satisfying the user are independent. That is, suppose two documents d_1 and d_2 are the results for a query q belonging to category c , the probability that the user will find *none* of the documents useful equals $(1 - V(d_1|q; c))(1 - V(d_2|q; c))$. Note that the assumption does *not* apply when the two documents belong to different categories.

3.2 Problem Formulation

Let us first hone our intuition about why a simple scheme that proportionately allocates the number of results to show for each category according to the percentage of users interested in that category may perform poorly. Consider the example of Flash, and suppose the intent distribution assigns a probability of 0.6 that the query is related to Technology. The proportional scheme would suggest that we show six documents out of ten about the "Adobe Flash player". However, having pointed the user to the official Adobe site as the first result, do we really need to show another five different sites? The missing link in this simple scheme is that the attractiveness of adding another document decreases when we have already shown a high quality document from the same category.

We now state the problem of *result diversification*. Suppose users only consider the top k returned results of a search engine. Our objective is to maximize the probability that the average user finds *at least* one useful result within the top k results.

Diversify(k): Given query q , a set of documents D , a probability distribution of categories for the query $P(c|q)$, the quality values of the documents $V(d|q; c)$, and an integer k . Find a set of documents $S \subseteq D$ with $|S| = k$ that maximizes

$$P(S|q) = \sum_c P(c|q) \left(1 - \prod_{d \in S} (1 - V(d|q; c)) \right) \quad (1)$$

Let us now interpret the objective in Equation 1 and see how it formalizes our intuition. Recall that $V(d|q; c)$ can be interpreted as the probability that a document d satisfies a user that issues query q with the intended category c . The value $(1 - V(d|q; c))$ is then the probability that d fails to satisfy. For a given category c , therefore, the probability that the set of documents will *all* fail to satisfy equals its product, by the aforementioned independence assumption. One minus that product equals the probability that some document will satisfy category c . Finally, summing up over all categories, weighted by $P(c|q)$, gives the probability that the set of documents S satisfies the "average" user who issues query q .

Note that this objective addresses the problem we mentioned in the beginning of the section. Specifically, if we have a very good document for "Flash" (query q) in the technology section (category c), like "Adobe Flash player" (document d), then $V(d|q; c)$ will be very high. The gains from other technology documents in reducing the probability that all documents fail to satisfy the user is very small, so a set of documents that does well under our objective will not expend more effort in the technology section.

Note also that our objective is not about including as many categories as possible in our result set; it is possible that even if there are less than k categories, not all categories will be covered. This is because our formulation explicitly considers how well a document satisfies a given category through $V(d|q; c)$. Hence, if a category c that is a dominant interpretation of the query q is not satisfied adequately, more documents from category c will be selected, possibly at the expense of not showing certain categories altogether.

As stated, the problem Diversify(k) does not take into account the *ordering* of the results, since the objective function is defined over (unordered) set of documents. This is due to the assumption that users will consider all k results. In practice, the precise value of k is of course not known in advance, i.e., different users may stop at different number of results. One may instead want to formulate an objective that takes into account the distribution of users who will stop at different values of k . Unfortunately, such distribution is not usually available. Nonetheless, to account for the importance of ordering, in our evaluation, our metrics are weighted to assign more value to documents higher in the ordering. Our algorithm is also designed to generate an ordering of results rather than just a set of results.

3.3 Complexity

Unfortunately, but not too surprisingly, the desired objective is NP-hard to optimize.

Lemma 1. Diversify(k) is NP-hard.

In fact, when documents belong to multiple categories, there may not exist a single ordering of documents such that the objective function of Diversify(k) is maximized for all k . This is because the set of documents optimal for Diversify($k-1$) need not be a subset of documents optimal for Diversify(k). Consider a two-category, three-document instance of Diversify. Suppose $P(c_1|q) = P(c_2|q) = 0.5$. Further, suppose the $V(d|q; c)$ values for the different categories are given according to Table 1. The optimal ordering for Diversify(1) is $d_1; d_2; d_3$, whereas it is $d_2; d_3; d_1$ for Diversify(2).

Table 1: Non-optimality of a single ordering

| Document | $V(djq; c_1)$ | $V(djq; c_2)$ |
|----------|---------------|---------------|
| d_1 | 0.8 | 0.8 |
| d_2 | 1.0 | 0.0 |
| d_3 | 0.0 | 1.0 |

3.4 Submodularity

Fortunately, not all is lost. The set function $P(Sjq)$ is in fact rich of structure, and admits a simple greedy strategy that will solve the problem quite well. The precise structure we are going to exploit is *submodularity* [12].

Definition 1 (Submodularity). *Given a finite ground set N , a set function $f : 2^N \rightarrow \mathbb{R}$ is submodular if and only if for all sets $S, T \subseteq N$ such that $S \subseteq T$, and $d \in N \setminus T$, $f(S + d) + f(T) \geq f(S) + f(T + d)$.*

Intuitively, a submodular function satisfies the economic principle of diminishing marginal returns, i.e., the marginal benefit of adding a document to a larger collection is less than that of adding it to a smaller collection.

Lemma 2. $P(Sjq)$ is a submodular function.

3.5 A Greedy Algorithm for Diversify(k)

We propose a natural greedy algorithm for computing a solution to Diversify(k) that will not only select a set of results but also order them as well. Let $R(q)$ be the top k documents selected by some classical ranking algorithm for the target query. Our algorithm will reorder $R(q)$ to maximize the objective $P(Sjq)$.

Let $U(cjq; S)$ denote the conditional probability that the query q belongs to category c , given that all documents in set S fail to satisfy the user. Initially, before any document is selected, $U(cjq; \cdot) = P(cjq)$. The algorithm selects output documents one at a time. At every step, it chooses the document that has the *highest marginal utility*, $g(djq; c; S)$, computed as the product of the conditional distribution of categories, $U(cjq; S)$, and the quality value of the document, $V(djq; c)$. This marginal utility can be interpreted as the probability that the selected document satisfies the user given that all documents that come before it fail to do so. At the end of the loop, the conditional distribution is updated to reflect the inclusion of the new document to the result set using Bayes rule.

3.6 Analysis of IA-Select

We now analyze the performance of IA-Select. We first show that it is in fact optimal in the case when each document can satisfy one category.

Theorem 1. IA-Select is optimal when $jC(d)j = 1$ for all $d \in R(q)$.

As the document selection process is independent of k , and that k only serves to determine when the algorithm stops, we have the following corollary:

Corollary 1. When $jC(d)j = 1$ for all $d \in R(q)$, IA-Select solves Diversify(k) optimally for all $k > 0$.

Algorithm 1 IA-Select

Input $k, q, C(q), R(q), C(d), P(cjq), V(djq; c)$
Output set of documents S

```

1:  $S = \emptyset$ ;
2:  $8c; U(cjq; S) = P(cjq)$ 
3: while  $|S| < k$  do
4:   for  $d \in R(q)$  do
5:      $g(djq; c; S) = U(cjq; S) V(djq; c)$ 
6:   end for
7:    $d = \text{argmax}_{d \in R(q)} g(djq; c; S)$  [ties broken arbitrarily]
8:    $S = S \cup \{d\}$ 
9:    $8c \in C(d); U(cjq; S) = (1 - V(djq; c))U(cjq; S) + V(djq; c)$ 
10:   $R(q) = R(q) \setminus \{d\}$ 
11: end while
12: return  $S$ 
```

In general, when documents may belong to multiple categories, IA-Select is no longer guaranteed to be optimal. This is hardly surprising as the problem is NP-hard. Reconsider the two-category, three-document instance of Diversify given in Section 3.3. The optimal solution to Diversify(2) is $d_2; d_3; d_1$, whereas IA-Select will return the order $d_1; d_2; d_3$.

However, even in the worst case, the error of IA-Select is bounded. This follows from the result by Nemhauser, Wolsey, and Fisher [12]:

Theorem 2. For a submodular set function f , let S^* be the optimal set of k elements that maximizes f . Let S^0 be the k -element set constructed by greedily selecting element one at a time that gives the largest marginal increase to f . Then $f(S^0) \geq (1 - 1/e)f(S^*)$.

Since our objective, $P(Sjq)$, is submodular, and our algorithm IA-Select is greedy in the same sense as stated in the theorem, we have:

Corollary 2. IA-Select is a $(1 - 1/e)$ -approximation algorithm for Diversify(k).

Recall that $V(djq; c)$ can be interpreted as the true probability that document d can satisfy category c for query q . Note that the above claims do not depend on how close $V(djq; c)$ approximates the desired probability. This is important as one must estimate $V(djq; c)$ in practice. The submodularity of the objective function, and the optimality and approximation guarantee of IA-Select remain true even for arbitrary $V(djq; c)$.

3.7 Illustrative Description of the Algorithm

We describe the algorithm using an example. Consider a query q with two categories c_1 and c_2 with $P(c_1jq) = 0.7$ and $P(c_2jq) = 0.3$. Assume the result set $R(q) = \{d_1; d_2; \dots; d_{10}\}$, with quality values given in Table 2. We run the algorithm to compute the top 5 results. We trace the run below.

Initially, $U(c_1jq; \cdot) = P(c_1jq) = 0.7$ and $U(c_2jq; \cdot) = P(c_2jq) = 0.3$. We order all the documents in $R(q)$ according to their marginal utility. We find d_1 with $g(d_1jq; c_1; \cdot) = 0.35$ from category c_1 is the document with the highest marginal utility and is therefore added to S . After adding d_1 the conditional probability $U(c_1jq; \cdot)$ is updated to 0.35. The next document to add would be any of $d_8; d_9; d_{10}$ from c_2 since all of them have the next highest marginal

Table 2: Input to illustration of algorithm

| Document | $V(jq; c_1)$ | $V(jq; c_2)$ |
|----------------------|--------------|--------------|
| d_1 | 0.50 | 0.00 |
| d_2 | 0.20 | 0.00 |
| d_3 | 0.15 | 0.00 |
| $d_4; d_5; d_6; d_7$ | 0.05 | 0.00 |
| $d_8; d_9; d_{10}$ | 0.00 | 0.33 |

utility of 0.1. Say, we pick d_8 , since ties are broken arbitrarily. After adding d_8 to S , the conditional probability $U(c_2jq; fd_1; d_8g)$ is updated to 0.20. Next, we fill the remaining three slots on the page. We compute the marginal utility of all the remaining documents in $R(q)$ following which we select d_2 (from category c_1) because it has the highest marginal utility of 0.07 and we include this document into S and update $U(c_1jq; fd_1; d_8; d_2g)$ to 0.28. Proceeding this way, we include d_9 followed by d_{10} . Thus, we produce $S = fd_1; d_8; d_2; d_9; d_{10}g$ with clearly more documents from category c_2 even though there are more documents from category c_1 in $R(q)$.

4. PERFORMANCE METRICS

We first briefly review some classical IR metrics, namely NDCG, MRR, and MAP [11]. These metrics are widely used for measuring search quality. It has been observed, however, that they do not properly take into account the value of diversification [5, 6]. Several metrics have been proposed for evaluating the diversity of search results, including the %no metric [16], the k -call metric [6], and β -NDCG [7]. However, these metrics do not take into account how well a document satisfies a topic, and they do not consider the relative importance of different topics. We therefore propose generalizations to classical IR metrics that address these shortcomings.

4.1 Classical Measures: NDCG, MRR, MAP

Given a ranked result set of documents Q and an ideal ordering of the same set of documents R , the *discounted cumulative gain* (DCG) at a particular rank threshold k is defined as

$$DCG(Q; k) = \sum_{j=1}^k \frac{2^{r(j)} - 1}{\log(1 + j)};$$

where $r(j)$ is the judgment (0=Bad, 1=Fair, 2=Good, 3=Excellent) at rank j in set Q . The ideally ordered set R contains all documents rated for the given query sorted descending by the judgment value. Now the *normalized discounted cumulative gain* (NDCG) at a particular rank threshold k is defined as

$$NDCG(Q; k) = \frac{DCG(Q; k)}{DCG(R; k)};$$

NDCG discounts the contribution of a document to the score as its rank increases. Higher NDCG values correspond to better correlation with human judgments. NDCG value at rank threshold k when the set Q is clear from the context is often written as $NDCG@k$.

The *reciprocal rank* (RR) is the inverse of the position of the first relevant document in the ordering. In the presence of a rank-threshold T , this value is 0 if there is no relevant

document in positions below this threshold. The *mean reciprocal rank* (MRR) of a query set is the average reciprocal rank of all queries in the query set.

The *average precision* of a ranked result set is defined as

$$\frac{\sum_{j=1}^k P(j) \cdot Relevance(j)}{\sum_{j=1}^k Relevance(j)};$$

where j is the position of the document, $Relevance(j)$ denotes the relevance of the document in position j , and $P(j) = \frac{1}{\sum_{i=1}^j Relevance(i)}$. Typically, a binary value for $Relevance(j)$ is used by setting it to 1 if the document in position j has a human judgment of Fair or better and 0 otherwise. The *mean average precision* (MAP) of a query set is the mean of the average precisions of all queries in the query set.

4.2 Intent Aware Measures

Classical IR metrics focus solely on the relevance of documents. Consider a query that belongs to two categories c_1 and c_2 . Suppose $P(qjc_2) = P(qjc_1)$, and we have two documents, d_1 rated Excellent for c_1 (but unrelated to c_2), and d_2 rated Good for c_2 (but unrelated to c_1). All three metrics above will assign a higher score to the ordering $(d_1; d_2)$, yet the "average" user will find the ordering $(d_2; d_1)$ more useful. Our generalization factors in the user intents.

4.2.1 NDCG-IA

In the classical NDCG computation, we compute the ratio of the DCG value of a given ordering of results for a query with the DCG value of the *ideal* ordering of the same results. However, when a query may be of multiple intents, the ideal is bound to be different depending on the intents the results are evaluated against. Given a distribution on the categories for a query $P(cjq)$ and an category label on each document, we can compute the *expected* NDCG for a given ordering of results. For each intent of the query, we treat any document that does not match the intent as Bad and compute its intent-dependent $NDCG(Q; kjc)$. This is then aggregated by averaging into $NDCG-IA(Q; k)$. Formally,

$$NDCG-IA(Q; k) = \sum_c P(cjq) NDCG(Q; kjc);$$

Note that NDCG-IA captures many of the same properties that make NDCG useful for evaluating IR systems [10]. It takes into account the degree of relevance of the documents through $r(j)$, and the importance of ordering through discounting. When documents are all of single intent, to maximize NDCG-IA, these documents should be ordered in decreasing level of relevance relative to each intent. The main difference is that NDCG-IA takes into account the distribution of intents, and forces a tradeoff between adding documents with higher relevance scores and those that cover additional intents. It can be interpreted as what the "average" user will think of the NDCG of the given ordering of the results. A drawback of NDCG-IA is that it may not lie in $[0; 1]$. Rather, it is upper-bounded by some constant less than 1 unless a single ordering is perfect for all intents. We do not know of a way to determine this bound other than exhaustive search.

We illustrate the computation of $NDCG-IA@5$ by reworking the example from Section 3.7. The Results Ordering column in Table 3 shows the ordering produced by IA-Select. We have additionally shown the judgment value for each

Table 3: Illustration of NDCG-IA@5 calculation

| | | a_1 | | a_2 | |
|-----------|------------------|---------------|------------|---------------|---------------|
| Pos | Results Ordering | Results | Ideal | Results | Ideal |
| 1 | $(d_1, 4)$ | $(d_1, 4)$ | $(d_1, 4)$ | $(d_1, 0)$ | $(d_8, 3)$ |
| 2 | $(d_8, 3)$ | $(d_8, 0)$ | $(d_2, 4)$ | $(d_8, 3)$ | $(d_9, 2)$ |
| 3 | $(d_2, 4)$ | $(d_2, 4)$ | $(d_3, 3)$ | $(d_2, 0)$ | $(d_{10}, 2)$ |
| 4 | $(d_9, 2)$ | $(d_9, 0)$ | $(d_4, 2)$ | $(d_9, 2)$ | $(-, 0)$ |
| 5 | $(d_{10}, 2)$ | $(d_{10}, 0)$ | $(d_5, 2)$ | $(d_{10}, 2)$ | $(-, 0)$ |
| 6 | $(d_3, 3)$ | | | | |
| 7 | $(d_4, 2)$ | | | | |
| 8 | $(d_5, 2)$ | | | | |
| 9 | $(d_6, 0)$ | | | | |
| 10 | $(d_7, 0)$ | | | | |
| DCG@5 | N/A | 22.50 | 30.42 | 6.87 | 10.39 |
| NDCG@5 | N/A | 0.7396 | | 0.6612 | |
| NDCG-IA@5 | N/A | 0.7161 | | | |

document in this column. Columns a_1 and a_2 show the results and ideal orderings up to position 5 along with judgment values. Note that the results ordering does not change, but the judgment value for some of the documents changes depending on the category of the query and the document. In the Results column under a_1 , d_8 , d_9 , and d_{10} receive a 0 judgment value since they belong to c_2 . Similarly, in the Results column under a_2 , d_1 and d_2 receive 0 judgment scores. At the bottom of the table, we show the resulting DCG scores for each intent, followed by a row for the NDCG values. Finally, we show the NDCG-IA score below the combined a_1 and a_2 column set. Recall from Section 3.7 that $P(c_1jq) = 0.7$ and $P(c_2jq) = 0.3$. So the final calculation is:

$$\text{NDCG-IA}(Q;5) = (0.7 \cdot 0.7396) + (0.3 \cdot 0.6612) = 0.7161$$

4.2.2 MRR-IA and MAP-IA

Using the same idea of averaging over user intent, we define intent aware MRR to be

$$\text{MRR-IA}(Q; k) = \frac{1}{c} \sum_{j \in c} P(cjq) \text{MRR}(Q; kjc):$$

Similarly, intent aware MAP is defined to be

$$\text{MAP-IA}(Q; k) = \frac{1}{c} \sum_{j \in c} P(cjq) \text{MAP}(Q; kjc):$$

5. EMPIRICAL EVALUATION

We evaluate our approach for result diversification against three commercial search engines. To preserve anonymity, we refer to these as Engines 1, 2, and 3.

We conduct three sets of experiments, differing in how the distributions of intents and how the relevance of the documents are obtained. In the first set of experiments, we obtain the distributions of intents for both queries and documents via standard classifiers, and the relevance of documents from a proprietary repository of human judgements that we have been granted access to. This allows us to conduct a large-scale study of how IA-Select performs compared to the other engines, but the results must be taken with a grain of salt as the input to IA-Select and the evaluation of the metrics uses the same intent distributions.

In the second set of experiments, we obtain the distributions of intents for queries and the document relevance using the Amazon Mechanical Turk platform (www.mturk.com).

This experiment is intended as a sanity check against the results obtained in the first one, especially with respect to bias introduced due to the sharing of intent distributions between IA-Select and the metrics. More specifically, while IA-Select still uses the intent distribution from the classifier aforementioned, the metrics are now evaluated with respect to the distributions of intents and the relevance scores obtained from the human subjects.

In the third set of experiments, we use a hybrid of data sources for evaluation | the distribution of intents for queries are from the Mechanical Turk platform, whereas document relevance is from the repository used in the first experiment. This allows us to focus on how the performance of IA-Select depends on having the exact knowledge of the query intent distribution.

In all of the experiments, intents are classified according to the ODP taxonomy (www.dmoz.org). Documents are scored for relevance on a 4-point scale as described in Section 4. The results used in the evaluation of the commercial search engines are based on the actual documents returned for the queries issued. The results generated by IA-Select is based on the following input: intent distribution obtained from a query classifier; set of documents given by the top 50 results from one of the search engines; intent-specific quality value given by the relevance score obtained from the same search engine, multiplied by the probability that the document belongs to the corresponding category determined by a document classifier. The ordering generated by IA-Select can be interpreted as an attempt to re-order the results from that search engine to take into account diversification under our problem formulation. One can also view the results as a comparison between a version of IA-Select that is ignorant of intents and one that takes into account intents.

For evaluation, we employ the three intent-aware metrics explained in Section 4. An alternative is to conduct side-by-side studies that try to capture user satisfaction more directly. Our choice is dictated by the fact that side-by-side comparisons are usually highly noisy and hence require large sample sizes, and as a result limits the scope of our study. On the other hand, the metrics we use can leverage human judgments already present in the repository, and gives us the flexibility to use different distribution of intents in our evaluation.

In these studies, we have chosen to use proprietary datasets

Figure 1: Distribution of categories (All experiments).

Figure 2: Distribution of human judgments (Experiments 1 and 3).

as we could not find public-domain ones that have sufficient data on query and document classification for training and evaluation. We believe it will be worthwhile to develop such datasets. Some of the TREC tracks (e.g. Novelty, Robust Retrieval) might provide good starting points.

5.1 Experiment 1: Proprietary Dataset

5.1.1 Setup

In this experiment, we obtain access to a dataset with 10,000 random queries with the associated top 50 documents returned from the three commercial search engines. For many of these documents, especially the ones in the top 10 result of some search engine, they are assigned human judgments as to their relevance to the query. As mentioned, the queries and the documents are classified according to the ODP taxonomy. Queries are classified using the algorithm described in [1], whereas documents are classified using a Rocchio classifier [11].

From this dataset, we sample about 900 queries conditioned on (a) the query has been classified to at least two categories, and (b) a significant fraction of associated documents have human judgments. The resulting distribution of the number of categories per query is shown in Figure 1. Of the documents from this sample, 85% of the documents have human judgments. The distribution of judgments across all documents is shown in Figure 2.

In evaluating the different approaches, results may contain unjudged documents. This is very rare for results from the commercial search engines (< 10%), but present a problem

Figure 3: NDCG-IA values (Experiment 1).

Figure 4: MAP-IA values (Experiment 1).

for IA-Select. To account for these documents, we sampled their labels from the distribution presented in Figure 2, the rationale being that these documents were not random documents of unknown quality but the top results of one of the search engines. Due to this sampling procedure, our experiments are not deterministic. In our evaluation, we sample labels five times and compute the mean results. All reported results have *non-overlapping* 95% confidence intervals, the exceptions being NDCG-IA@3 and NDCG-IA@5 results for Engines 2 and 3.

5.1.2 NDCG-IA Results

The results for NDCG-IA are shown in Figure 3. Results produced by IA-Select have been labeled Diverse. Note that the NDCG values are conventionally quoted by multiplying fractional values by 100 and achieving even one point gain is considered difficult. We caution that the y-axis on this and subsequent figures do not start at zero; we have chosen the scale so as to provide sufficient resolution to distinguish among the performances of the four approaches.

The overall trend is clear. At all rank thresholds we evaluated, the orderings produced by IA-Select are significantly better than all three engines. Furthermore, as higher thresholds are considered, the outperformance of IA-Select increases. This widening of the outperformance margin as rank increases can be explained by the fact that IA-Select makes use of the extra search results to cover a wider range of categories of the query.

5.1.3 MAP-IA and MRR-IA Results

The results for MAP-IA and MRR-IA are shown in Figures 4 and 5 respectively. IA-Select outperforms all three

Table 4: NDCG-IA values (Experiment 2)

| NDCG-IA | Search Engine | Diverse |
|---------|---------------|---------|
| @1 | .8629 | .8798 |
| @2 | .8501 | .8720 |
| @3 | .8577 | .8676 |
| @4 | .8614 | .8663 |
| @5 | .8603 | .8690 |

Figure 5: MRR-IA values (Experiment 1).

engines significantly under both metrics.

We observe that the precision scores for MAP-IA decrease as the rank threshold increases for all engines including IA-Select. This is due to additional relevant documents being presented out of order in later positions. Note that if no additional relevant documents were found past position 3, MAP-IA@5 and MAP-IA@10 would be equal to MAP-IA@3. This observation also holds for the classical MAP metric.

5.2 Experiment 2: Mechanical Turk Judgments

One cause of concern with Experiment 1 is that we have used the same distribution of intents $P(cjq)$ both as input to IA-Select and for evaluation. Does the outperformance of IA-Select rely entirely on this sharing of intents? To address this concern, we conduct a second experiment in which the distribution of intent used in computing NDCG-IA value is different from the input to IA-Select. We use the Amazon Mechanical Turk platform for obtaining human judgments in this study.

5.2.1 Setup

We sample 200 queries from the dataset used in Experiment 1 such that each query had at least three categories associated with it. We submit these queries along with the three most likely categories as estimated by the classifier to the Turks, and the top five results produced by IA-Select and one of the search engines in some random order. The Turks are asked to first choose a category (out of the three) they most closely associate with the given query. They then judge the corresponding results *with respect to the chosen category* using the same 4-point scale. They could label a result as N/A when they could not conclude the relevance of the document to the query. In such cases, we assigned bad label to the document. They could also decide not to judge a query if the query was out of their realm of expertise. For each query, we obtain evaluations from seven Turks.¹

Due to the use of the classifier in generating the candidate categories for the query in the tasks, one may worry that the distribution of intents obtained in this study are skewed towards the categories chosen by the classifier. To mitigate this concern, one can ask the subjects to choose a category from the ODP taxonomy, but we believe this will place too large a cognitive burden and may even generate

¹Some of the study parameters (e.g. number of Turks, number of categories per query, number of results per query, number of search engines) were constrained by the various factors such as the limitations imposed by the Mechanical Turk platform, our desire to keep the instructions to the Turks simple and concise, and the budget for the study.

Figure 6: Distribution of human judgments (Experiment 2).

meaningless results. Note that given some queries belong to four or more categories, this bias does not necessarily work in our favor. Nonetheless, we flag this as a potential concern in interpreting the results that followed.

5.2.2 Results

We found that different Turks selected different categories for more than 70% of the queries. We used this data to estimate $P(cjq)$ values when computing NDCG-IA metric. Note that the $P(cjq)$ used by IA-Select to diversify results was the same as used in Experiment 1 (obtained using the algorithm described in [1]).

Table 4 summarizes the results. The ordering produced by IA-Select (column titled Diverse) yields a better NDCG-IA score than the ordering produced by the search engine at all rank thresholds. The gap at threshold 1 suggests that the search engine did not place the document from the dominant query category at position 1. The widening of the gap at threshold 2 is quite interesting. We believe this is because search engines often show two documents from the same host in the first two positions, albeit they indent the second document. This does exactly the opposite of diversification. For queries with multiple intents, Turks often did not like such second documents.

We also observe that the NDCG scores in Table 4 are much higher compared to those in Figure 3. To understand this, we chart the distribution of judgments from this experiment in Figure 6. We can see that judgment scores are significantly higher for the documents considered in this experiment compared to the original distribution in Figure 2.

5.3 Experiment 3: Hybrid Evaluation

Having obtained $P(cjq)$ values from the user study reported in Experiment 2, we examined the impact of using them for the purposes of computing the performance metrics. Note that the $P(cjq)$ used by IA-Select to diversify results continued to be the same as used in Experiment 1.

Figure 7: NDCG-IA values (Experiment 3).

We thus now have a $P(cjq)$ for computing the performance of IA-Select which is different from the one it uses for diversifying results.

Figure 7 shows the results. We include the results only for the NDCG-IA metric; the trends are similar for the other two metrics. We use only those queries for which we could obtain $P(cjq)$ values from the user study. We observe that IA-Select again outperforms the three search engines, showing the importance of diversifying search results from the point of view of users.

It is tempting to compare Figure 7 with Figure 3. However, several cautions are in order. First, Figure 7 includes computation for queries that are a subset of those used for computing Figure 3. They are the ones that had at least three intents. Furthermore, we retained only top three intents for each query when we obtained user judgments.

Subject to above caveats, we can analyze why the NDCG-IA values are relatively lower in Figure 7. We are penalizing the scores of those queries that might have had more than three intents. Additionally, the NDCG-IA values for IA-Select are expected to be lower due to mismatch between the estimates for $P(cjq)$ obtained from two different sources. IA-Select still emerges as the winner.

6. CONCLUSIONS

We studied the question of how best to diversify results in the presence of ambiguous queries. This is a problem that most search engines face as users often underspecify their true information needs. We took into consideration both the relevance of the documents and the diversity of search results and presented an objective that directly optimizes for the two. We provided a greedy algorithm for the objective with good approximation guarantees. To evaluate the effectiveness of our approach, we proposed generalizations of well-studied metrics to take into account of the intentions of the users. Over a large set of experiments, we demonstrated that our approach consistently outperforms results produced by commercial search engines over all of the metrics.

The objective in Diversify can be viewed as a conservative metric that aims to maximize the probability that the average user will find some useful information among the search results. There are other reasonable objectives as well. For example, when users are looking for opinions on a product, a set of results is useful only when there are multiple results regarding the product. The objective function will need to be changed accordingly to take this into ac-

count. Another natural objective would be to minimize the expected rank at which the average user will find useful information. This is an important objective as the document ordering is known to have a huge impact on users' perception of how useful the results are. We plan to explore these alternatives in the future.

The true test of the performance of search engines is their ability to satisfy their users. Conventional metrics fail to accurately reflect the performance of different algorithms as they tend to ignore the presence of ambiguity in queries. We have taken a first step in addressing this shortcoming by introducing intent-aware metrics. We believe further work is necessary in designing new metrics that are more reflective of user satisfaction in their *interaction* with a search engine.

7. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their detailed review and thoughtful suggestions.

8. REFERENCES

- [1] Kannan Achan, Ariel Fuxman, Panayiotis Tsaparas, and Rakesh Agrawal. Using the wisdom of the crowds for keyword generation. In *WWW*, pages 1{8, 2008.
- [2] Aris Anagnostopoulos, Andrei Z. Broder, and David Carmel. Sampling search-engine results. In *WWW*, pages 245{256, 2005.
- [3] A. Bookstein. Information retrieval: A sequential learning process. *Journal of the American Society for Information Sciences (ASIS)*, 34(5):331{342, 1983.
- [4] B. Boyce. Beyond topicality: A two stage view of relevance and the retrieval process. *Info. Processing and Management*, 18(3):105{109, 1982.
- [5] Jaime G. Carbonell and Jade Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, pages 335{336, 1998.
- [6] Harr Chen and David R. Karger. Less is more: probabilistic models for retrieving fewer relevant documents. In *SIGIR*, pages 429{436, 2006.
- [7] Charles L. A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Buttcher, and Ian MacKinnon. Novelty and diversity in information retrieval evaluation. In *SIGIR*, pages 659{666, 2008.
- [8] W. Goeman. A searching procedure for information retrieval. *Info. Storage and Retrieval*, 2:73{78, 1964.
- [9] Dorit Hochbaum, editor. *Approximation Algorithms for NP-Hard problems*. Springer, 1999.
- [10] K. Jarvelin and J. Kekalainen. IR evaluation methods for retrieving highly relevant documents. In *SIGIR*, pages 41{48, 2000.
- [11] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge Univ Press, 2008.
- [12] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Math. Programming*, 14:265{294, 1978.
- [13] F. Radlinski, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. In *ICML*, 2008. First presented at NIPS07 Workshop on Machine Learning for Web Search.

- [14] Filip Radlinski and Susan T. Dumais. Improving personalized web search using result diversification. In *SIGIR*, pages 691{692, 2006.
- [15] Erik Vee, Utkarsh Srivastava, Jayavel Shanmugasundaram, Prashant Bhat, and Sihem Amer-Yahia. Efficient computation of diverse query results. In *ICDE*, pages 228{236, 2008.
- [16] E. M. Voorhees. Overview of the trec 2004 robust retrieval track. In *TREC*, 2004.
- [17] Yunjie Xu and Hainan Yin. Novelty and topicality in interactive information retrieval. *J. Am. Soc. Inf. Sci. Technol.*, 59(2):201{215, 2008.
- [18] ChengXiang Zhai. *Risk Minimization and Language Modeling in Information Retrieval*. PhD thesis, Carnegie Mellon University, 2002.
- [19] ChengXiang Zhai, William W. Cohen, and John D. Laerty. Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *SIGIR*, pages 10{17, 2003.
- [20] ChengXiang Zhai and John D. Laerty. A risk minimization framework for information retrieval. *Info. Processing and Management*, 42(1):31{55, 2006.
- [21] Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. Improving recommendation lists through topic diversification. In *WWW*, pages 22{32, 2005.

APPENDIX

A. PROOFS FOR SECTION 3

Lemma 1. Diversify(k) is NP-hard.

Proof. This follows from a reduction from Max Coverage, a NP-hard problem related to Set Cover [9]. In the Max Coverage problem, one is given a universe of elements U , a collection C of subsets of U , and an integer k . The objective is to find a set of subsets $S \subseteq C$, $|S| = k$, to maximize the number of covered elements.

Given an instance of the Max Coverage problem, we map U to the set of categories C . The collection of subsets C is mapped to the set of documents D , where if $C \subseteq C$ contains elements $fu_1; u_2; \dots; u_kg$, the corresponding document $d \in D$ can satisfy categories $fc_1; c_2; \dots; c_kg$, i.e., $V(djq; c_i) = 1$ for c_1 through c_k . One can easily verify that the optimal solution to Diversify(k) is optimal for Max Coverage. In fact, Diversify(k) is a *soft* generalization of Max Coverage, in the sense that the documents can partially satisfy a category. \square

Lemma 2. $P(Sjq)$ is a submodular function.

Proof. Intuitively, the function is submodular because a larger set of documents would have already satisfied more users, and therefore the incremental gain for an additional document is smaller. Let us now capture the intuition mathematically. Let $S; T$ be two arbitrary sets of documents related by $S \subseteq T$. Let e be a document not in T . Denote $S \cup \{e\}$ by S^0 and similarly $T \cup \{e\}$ by T^0 .

$$\begin{aligned} & P(S^0jq) - P(Sjq) \\ &= \sum_{c \in C} P(cjq) \left(\prod_{d \in S^0} (1 - V(djq; c)) \right) - \sum_{c \in C} P(cjq) \left(\prod_{d \in S} (1 - V(djq; c)) \right) \\ &= \sum_{c \in C} P(cjq) \left(\prod_{d \in S^0} (1 - V(djq; c)) \right) - \sum_{c \in C} P(cjq) \left(\prod_{d \in S} (1 - V(djq; c)) \right) \end{aligned}$$

$$= \sum_{c \in C} P(cjq) \left(\prod_{d \in S^0} (1 - V(djq; c)) \right) - \sum_{c \in C} P(cjq) \left(\prod_{d \in S} (1 - V(djq; c)) \right)$$

Similarly, we can establish that

$$P(T^0jq) - P(Tjq) = \sum_{c \in C} P(cjq) \left(\prod_{d \in T^0} (1 - V(djq; c)) \right) - \sum_{c \in C} P(cjq) \left(\prod_{d \in T} (1 - V(djq; c)) \right)$$

However, note that for all c ,

$$\prod_{d \in T^0} (1 - V(djq; c)) \leq \prod_{d \in T} (1 - V(djq; c)) \leq \prod_{d \in S} (1 - V(djq; c)) \leq \prod_{d \in S^0} (1 - V(djq; c))$$

Therefore, we conclude that

$$P(S^0jq) - P(Sjq) \leq P(T^0jq) - P(Tjq)$$

as desired, i.e., the function $P(Sjq)$ is submodular. \square

Theorem 1. IA-Select is optimal when $jC(d)j = 1$ for all $d \in R(q)$.

Proof. First, note that if each document can only be of only one category, i.e., $jC(d)j = 1$, $g(djq; c; S)$ is solely determined by $U(C(d)jq; S)V(djq; C(d))$. Therefore, for a given category c , the ordering of the documents $R(q)$ with respect to $g(djq; c; S)$ is the same for all S . For a set of documents S , by replacing the documents in each category with the ones with the highest $V(djq; c)$, $P(Sjq)$ can only improve. Let us call this *local improvement*. Note that the results returned by IA-Select cannot be locally improved, since it selects only the documents with the highest $V(djq; c)$ in each category. Suppose contrary to our claim, there exists a solution T that is strictly better the result S returned by that IA-Select. We can first carry out locally improvements of T as specified. If S and T has the same number of documents in a category, then these documents must be identical. Therefore, for T to be better, it must include different number of documents for some categories.

Given that $jSj = jTj$, if T has more results than S in category c_1 , then there must be some other category c_2 for which S has more results. Consider the worst document, d_1 , that T returned for category c_1 , and the best document, d_2 , that S returned for category c_2 that T does not include. Let $S^0 = S \setminus \{d_1\} \cup \{d_2\}$ and $T^0 = T \setminus \{d_1\}$.

Since the algorithm is greedy, the marginal benefit of adding d_2 to S^0 must be as high as that of adding d_1 , or d_1 would have been chosen instead of d_2 . By submodularity, compared to S^0 , the marginal benefit of adding d_2 can only be higher for T^0 , since T^0 has only a subset of documents in category c_2 , and the marginal benefit for adding d_1 can only be lower, since T^0 has a superset of documents in category c_1 . Therefore,

$$\begin{aligned} P(T^0 \cup \{d_2\}jq) - P(T^0jq) &\leq P(S^0 \cup \{d_2\}jq) - P(S^0jq) \\ &\leq P(S^0 \cup \{d_1\}jq) - P(S^0jq) \\ &\leq P(T^0 \cup \{d_1\}jq) - P(T^0jq) \end{aligned}$$

Hence, $P(T^0 \cup \{d_2\}jq) - P(T^0 \cup \{d_1\}jq) = P(Tjq)$. Note that $(T^0 \cup \{d_2\})$ is more similar to set S than T was, i.e., $jS \setminus Tj < jS \setminus (T^0 \cup \{d_2\})j$. By repeating this argument, we eventually obtain a chain of inequalities that lead to S , which yields $P(Sjq) - P(Tjq)$, contradicting the supposition that T is better. \square