Diversity in Ranking via Resistive Graph Centers

Avinava Dubey
IBM Research India
avinava.dubey@gmail.com

Soumen Chakrabarti
IIT Bombay
soumen@cse.iitb.ac.in

Chiranjib Bhattacharyya
IISc Bangalore
chiru@csa.iisc.ernet.in

ABSTRACT

Users can rarely reveal their information need in full detail to a search engine within 1–2 words, so search engines need to "hedge their bets" and present diverse results within the precious 10 response slots. Diversity in ranking is of much recent interest. Most existing solutions estimate the marginal utility of an item given a set of items already in the response, and then use variants of greedy set cover. Others design graphs with the items as nodes and choose diverse items based on visit rates (PageRank). Here we introduce a radically new and natural formulation of diversity as finding centers in resistive graphs. Unlike in PageRank, we do not specify the edge resistances (equivalently, conductances) and ask for node visit rates. Instead, we look for a sparse set of center nodes so that the effective conductance from the center to the rest of the graph has maximum entropy. We give a cogent semantic justification for turning PageRankthus on its head. In marked deviation from prior work, our edge resistances are learnt from training data. Inference and learning are NP-hard, but we give practical solutions. In extensive experiments with subtopic retrieval, social network search, and document summarization, our approach convincingly surpasses recently-published diversity algorithms like subtopic cover, max-marginal relevance (MMR), GRASSHOPPER, DIVRANK, and SVMDIV.

Categories and Subject Descriptors

H.3 [**Information storage and retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Performance

Keywords

graph, conductance, diversity, ranking

1. INTRODUCTION

Learning to rank is an active area of machine learning and data mining research [18]. Queries are short (1–2 words) and are often an incomplete excerpt of the user's information need. For example, many person names are ambiguous. User attention drops off steeply with rank, and they rarely look beyond the first 10 or 20 hits.

Together, these two phenomena make *diversity* a critical requirement. The goal of the search engine is to "hedge its bets" and present a variety of response items within scarce screen real estate (10–20 top-ranking positions), so as to minimize the (expected) number of users who abandon the search without satisfying their information need.

Depending on the application, diversity of response items may be interpreted in different ways. For person name queries on the Web, search engines usually return home pages of different people sharing the name. For queries related to broad topics, diversity may mean adequate coverage of subtopics. For commerce search over laptops or cameras, users may expect a faceted or tabular view of models and attributes. In extractive document summarization, a minimal amount of non-repetitive text must be extracted from given documents. Here the items are typically sentences.

1.1 Prior art

The diversity motive clearly requires a global item selection strategy, because the desirability of including an item in a query response obviously depends on other items, even if its intrinsic relevance does not. This central observation has led to two major approaches to diversity.

In the *cover* approach, the marginal worth of including an item in the response is evaluated as some function of the information it contains that is not elsewhere in the response. Subtopic coverage [29], max-marginal relevance (MMR) [4] and submodular coverage [17, 16] are examples of this paradigm where the marginal utility is designed by hand. SVMDIV [28] and INDSTRSVM [15] learn the marginal utility of subtopic coverage of documents from training data.

In the *Markov walk* approach, a graph is designed with nodes representing items. Weighted edges represent some form of similarity between items that is designed by hand, using domain knowledge. The resulting graph is often called a *resistive* graph, with edges having resistance (equivalently, *conductance* representing transition probabilities). Random walk processes are defined on the graph. Nodes are included in the response based on visit rates [19] or expected numbers of visits before absorption [32].

A recent, third approach to diversity uses proprietary click-through data [2] from search engines, sometimes in online settings [23]. The basic idea is that if clusters of users click to different pages after querying a person name, we know that future responses to the query should be diversified. For this approach to succeed, enough users should see diverse URLs to click in the first place, which means that "cold-start diversity" remains an important problem.

Here we focus on the first two approaches, and describe the most closely related work in more detail in Section 2.

1.2 Our contributions

Our work began by recognizing some limitations in the cover and Markov walk paradigms above.

- In the cover approach, at test time, how one item covers (thus rendering redundant) another item is not directly visible. Therefore, one cannot create features for the units of coverage, even if these are known during training (as in subtopic queries).
- The Markov walk approach can potentially get around

KDD'11, August 21–24, 2011, San Diego, California, USA.

this problem by directly comparing items without reference to the units of coverage. However, existing work hardwires edge weights rather than learn them.

- Markov walks express *associativity* [1]: a node i with large score linking to node j pulls up the score of j. However, diversity demands *dissociative* decisions: if i and j are both relevant and very similar, we should pick only one of them.

The last-mentioned disconnect between the semantics of diversity and the Markov walk approach has led to somewhat contrived fixes that we will discuss in Section 2.

Our main contribution is GCD (graph center diversity), a new framework for diversity. Roughly speaking, GCD finds a sparse teleport [12] for an associative Markov walk to have a high-entropy stationary distribution. It is semantically well-motivated and more effective in practice than existing algorithms. GCD is novel in several key ways:

- In standard Markov walks [12, 26], walk parameters, including the teleport vector, are inputs and node scores are the output. In contrast, in our new framework, the *teleport is the output*. We know of no other attempts to use Markov walks in this fashion. This critical role reversal allows us to reuse associative graphs for diversity, while avoiding their conceptual limitation.
- In Markov walk approaches, walk parameters are hand-designed and hardwired. In contrast, we learn the walk parameters from an expressive parameter space, using training data.
- Our framework incorporates attention decrease with rank ("presentation bias") in a natural way.

Somewhat unsurprisingly, inference in GCD is NP-hard. However, simple and practical heuristics work well. The framework is presented in Section 3. Inference is discussed in Section 4. Training the parameters is discussed in Section 5. The training problem has some resemblance to structured learning [27], but with additional complications. GCD has some similarity with maximizing diffusion of influence in networks [13] but the latter involves dynamic models. GCD is reminiscent of centerpiece subgraphs [25]. Their goal is not diversity, but salient hub nodes connecting a *small* set of query nodes, and walk parameters are not learnt.

In Section 6 we report on experiments with the TREC subtopic task, queries on the IMDB graph of movies, actors, and countries, and the DUC document summarization task. All data sets are public and have been used to evaluate recent algorithms. We show that the elegance of GCD is corroborated by substantially better diversity as evaluated by the end applications.

2. RELATED WORK

2.1 Marginal relevance

Carbonell *et al.* [4] were among the earliest to note the potential conflict between relevance and diversity and offer a trade-off. Given query q and a set of documents S already reported, maximum marginal relevance (MMR) proposes greedily choosing document $\arg\max_{d\not\in S} \lambda\,\mathrm{sim}_1(d,q) - (1-\lambda)\max_{d'\in S}\mathrm{sim}_2(d,d')$ as the next document, for suitably designed similarity functions $\mathrm{sim}_1, \mathrm{sim}_2$ and tuned parameter λ. Zhang *et al.* [31], Chen *et al.* [5] and Guo *et al.* [11] offer probabilistic and/or mixture model formulations addressing the same basic issue.

2.2 Subtopic coverage

Zhai *et al.* [29] proposed subtopic coverage as a diversification objective and demonstrated effective algorithms on the TREC interactive track. In the learning-to-rank literature, Yue and Joachims [28] proposed a structured learning framework SVMDIV for diverse topic coverage, by using features that capture word coverage signals as surrogates of topic coverage. INDSTRSVM [15] propose additional constraints to encourage diversity and balance appropriate for the specific application of summarization. SVMDIV and INDSTRSVM stand out as among very few diversity approaches that learn from a powerful hypothesis space.

2.3 Submodular subset selection

An approach related to MMR [17, 16] represents items as nodes V in a graph, edges with weights w_{ij} representing similarity, and seeks a subset $S \subset V$ to maximize $\sum_{i\in S, j\in V\setminus S} w_{ij}$ (or a similar function) while minimizing the redundancy or self-similarity $\sum_{i,j\in S, i\neq j} w_{ij}$. I.e., they maximize the first term minus λ times the second, as in MMR. Such objectives are *submodular*, affording approximation guarantees from simple greedy algorithms. Despite using a graph representation, this approach has no connection with associative Markov walks. Enhanced with item sizes c_i and a budget constraint $\sum_{i\in S} c_i \le B$, submodular subset selection has been shown to be particularly suited for document summarization. Similar subset selection has been used to summarize blogs [8]. That paper requires a fixed and known notion of how each blog posting covers subtopics, which precludes its application in our settings. It also proposes an online algorithm to learn preference weights of individual users on the fixed subtopics, but this has no connection with our learning algorithm.

2.4 Random walk variations

Closest to our proposal here are the following papers, each of which represents the documents to be (re)ranked as nodes in a graph with edges representing similarity.

Zhang *et al.* [30] design an "affinity graph" between items (documents) to be ranked. Suppose $C(j|i)$ is the probability of transition from item i to j. $C(j|i)$ is large if most words in i are also in j, reducing the novelty of i. They first compute PageRank on the affinity graph. The initial scores of items are set to their PageRanks. Then, in a loop, they remove the document j with the largest score $s(j)$, and reduce the score of linked documents i by $s(j)\,C(j|i)$. Although it uses a PageRank computation, this approach is very similar in spirit to marginal relevance.

Zhu *et al.*'s GRASSHOPPER algorithm [32] also sets up a random walk with transition probabilities related to similarity between items. They find the top-ranking item using conventional personalized PageRank [12], but then make the corresponding node a *sink*, i.e., having no outbound transition edges. The resulting graph no longer has a meaningful stationary distribution (the PageRank of the sink state will be 1), so they compute the expected number of visits to each node before the random walk is absorbed into the sink node. The node with most visits is the second-ranked node. Then they make the second node a sink as well and continue the process. This is the graph analog to the $k = 1$ case of Chen and Karger [5, Section 7.1].

The latest example of this style of formulation is DIVRANK [19], which proposes a random walk with *time-variant* transition probabilities $C_T(j|i)$ at time T:

$$C_T(j|i) = (1-\lambda)r(j) + \lambda \frac{C_0(j|i)N_T(j)}{\sum_k C_0(k|i)N_T(k)}$$

where $N_T(j)$ is the (random) number of times node j has been visited up to time T and r is a multinomial teleport distribution. This translates into

$$p_{T+1}(j) = (1-\lambda)r(j) + \lambda \sum_i \frac{C_0(j|i)N_T(j)}{\sum_k C_0(k|i)N_T(k)} p_T(i)$$

Unfortunately $N_T(i)$ must be approximated with point estimates to keep the computation practical. Mei *et al.* argue that the above implements a "rich gets richer" effect as in preferential attachment [3]: through random choice and/or asymmetry in the graph neighborhood, one node will emerge the "winner" in each tightly connected subgraph. This strategy achieves the same end as Zhang *et al.*'s [30] discounting of information richness through the affinity graph.

Other uses of such random walks exist [6], but walk parameters are again hardwired, and the results are used to define similarity between items which is then used in the formulation.

Despite differences in specifics, much is in common across the above techniques [30, 32, 19]. In all cases a graph is defined with edges representing similarity, and a random walk is formulated with parameters that are predefined and hardwired. We argue that diversity has no semantic foundation in visit rates of random surfers and rich-gets-richer phenomena. *Diversity is not the outcome of a social process.* Diversity is a *desirable property of a ranking* that we should seek when query intent is uncertain.

2.5 Associative vs. dissociative graph models

If nodes (documents) are labeled relevant or irrelevant, all graphs defined above would have *associative edge potentials* in graphical model terminology [14], i.e., neighboring nodes will tend to have the same label. This is just a restatement of the cluster hypothesis in IR, and has been used for collective (associative) ranking using the undirected graph Laplacian [21]. Let a_{ij} be the similarity between items i, j, $b(i)$ a non-collective score wrt the query, and $f(i)$ the final score we seek. Laplacian techniques seek to minimize $\sum_{i,j} a_{ij}\big(f(i) - f(j)\big)^2 + \diamondsuit \sum_i \big(f(i) - b(i)\big)^2$, i.e., we want scores $f(i)$ to be "smooth" across edges while not differing much from the local scores $b(i)$. PageRank corresponds to a directed version of the Laplacian [1], but the edges continue to be associative.

In contrast, diversity calls for *dissociative* edge potentials. Let the node labels now be "report" vs. "do not report". Relevance is now a *node potential* [14] rather than a node label: in the absence of edges, relevant nodes should be reported and vice versa. But if an edge (i, j) represents a strong similarity and i, j are both relevant, then the diversity principle dictates that the labels across the edge be *different* rather than the same. It is therefore unsurprising that the above attempts to use an associative network in an essentially dissociative labeling task result in an uneasy fit.

While dissociative graphical models are a more natural expression of the diversity objective, they are computationally intractable [14]. In contrast, PageRank and Laplacian smoothing can be implemented efficiently via power iterations and quadratic optimization, respectively.

This raises the central question addressed in this paper: Can we retain the simplicity and efficiency of associative Markov walks while making dissociative ranking decisions?

3. DIVERSITY AS TELEPORT SEARCH

In this section we introduce the GCD diversity framework. First we list the notation and facts about PageRank used hereafter. In what follows, i, j represent nodes (interchangeably, items to be ranked) in a graph. Let the number of nodes in the graph be N.

- $C(j|i)$ is the probability of walking from node i to j. We will also write it as a matrix element $C(j, i)$ (note the transposition).
- $\alpha \in (0, 1)$ is the probability of walking, and $1 - \alpha$ is the probability of teleporting, at any step.
- When a teleport happens, the destination is node i with probability $r(i)$; r is a multinomial teleport distribution.
- The resulting PageRank vector is called $p(\alpha, C, r)$ and satisfies the recurrence $p(\alpha, C, r) = \alpha C p(\alpha, C, r) + (1 - \alpha)r$, which solves to $p(\alpha, C, r) = (1-\alpha)(\mathbb{I} - \alpha C)^{-1} r$, where $\mathbb{I}$ is the identity matrix.

We will denote $(1-\alpha)(\mathbb{I}-\alpha C)^{-1}$ as the the $N \times N$ matrix M.

3.1 From random surfer to teleported searcher

In standard (personalized) PageRank, α, C, r are fixed, and the search engine uses $p(\alpha, C, r)$ as an estimate of the prestige of pages. I.e., the actions of the random surfer determine the score and rank assigned by the search engine.

However, by now, the surfer's predominant access path into the Web is initiated by a search. Search engine rankings form a narrow peephole (typically $K = 10$ links at a time) through which users explore the Web. (Search engines greatly influence visit and link rates of Web pages. Increased visibility via high PageRank can lead to more visits and links, which can set up a feedback [9] and delay or prevent prominence of newcomer pages.)

3.1.1 Reversing the role of teleport r

In successfully using an associative graph model for dissociative ranking, our key insight is to regard teleport as the *output* of the search engine as against an *input* to its ranking processes. Suppose we regard the $K = 10$ links returned by the search engine in response to a query as defining a teleport vector. The user then uses these response links to teleport to some of these pages. From there the user can locate other similar pages using a variety of devices:

- Following explicit hyperlinks
- Acquiring vocabulary by reading documents and composing further queries
- Consulting topic directories and hubs to locate related pages

Thus, the search engine's initial response sets off a diffusion process through which the user experiences a subset of the corpus.

3.1.2 Omniscient view of relevance, b

The search engine, meanwhile, has analyzed tens of billions of pages and has an "omniscient" view of the relevance of a document to the user's query. Fixing the query, the omniscient relevance score of document i is denoted $b(i)$. In a typical linear scoring framework, this is just $x_{qi}^\top w$, where w is a scoring model vector and x_{qi} is a vector space representation of document i for query q.

In vector space cosine scoring, $x_{qi}^\top w$ is already scaled to lie between 0 and 1, with a score of 0 representing complete irrelevance. (If this does not hold for the raw scores, a variety of transformations, such as the logit function, can be used.) We rescale document scores so that the scores of all documents add up to 1, thereby making b a multinomial distribution. To summarize:

> The goal of the search engine is to choose teleport r so as to make the "teleported searcher" visit node i with probability close to $b(i)$.

3.2 Teleport profile: Modeling rank bias

To start with, we can assume that the user samples each of the K links provided by the search engine uniformly at random. Under this assumption the teleport vector r induced by the search engine output will have exactly K nonzero elements, each equal to $1/K$.

It is now established [10, 22] that user attention to ranked responses is heavily skewed toward top ranks. Accordingly, we can use a *decaying profile* for teleport r. We insist that the K nonzero values be $a_1, \ldots, a_K$ summing to 1, with $a_1 \geq \cdots \geq a_K$. Here are some profiles we evaluated.

Uniform: This is the simplest option, $a_k = 1/K$.

Exponential: $a_k \propto 2^{-k}$, motivated by clickthrough rates [10].

Reciprocal: $a_k \propto 1/k$, motivated by mean average precision and mean reciprocal rank [18].

Logarithmic: $a_k \propto 1/\log k$, motivated by the NDCG ranking accuracy measure [18].

In all cases, the K nonzero *values* in teleport r are predetermined; the search engine must find K nodes and order them from $1, \ldots, K$ so that the design of r can be completed.

As a single profile policy across all queries and data sets, the logarithmic profile worked consistently better than other profiles in our experiments. However, it is conceivable that different kinds of queries (e.g., navigational vs. informational) may benefit from different profiles. This is an interesting direction for future work.

3.3 Exploiting linearity and PPVs

As we have seen, given teleport r, conductance matrix C, and walk probability α, the PageRank vector $p = (1-\alpha)(\mathbb{I} - \alpha C)^{-1} r = Mr$, say. This means that p is a linear function of r. If we denote this functional dependency as p_r, we have $p_{cr} = c p_r$ for $c \in \mathbb{R}$, and $p_{r_1+r_2} = p_{r_1} + p_{r_2}$.

A special class of teleport vectors, called *impulse teleports*, reset to one node with probability 1. The impulse teleport to node i is written δ_i, where $\delta_i(i) = 1$ and $\delta_i(j) = 0$ for all $j \neq i$. The PageRank for δ_i, called the *personalized PageRank vector* for node i and denoted PPV_i, is simply $p(\delta_i)$.

Suppose we precompute PPV_i for all nodes i. PPV_i is the ith column of M, written M^i. Then, given an arbitrary teleport vector r, we can express $p_r = \sum_j M^j r(j)$, using the linearity property.

3.3.1 The divergence objective

The problem thus reduces to the following: Given multinomial probability vectors M^i, $i = 1, \ldots, N$ and scalars $a_1, \ldots, a_K$ adding up to 1, find indices $i_1, \ldots, i_K$ such that

$$\sum_{k=1}^{K} a_k M^{i_k} \approx b, \tag{1}$$

where b is the omniscient relevance. In place of "$\approx$" above, in an implementation, we would seek to

$$\min_{i_1,\ldots,i_K} \left\| b - \sum_{k=1}^{K} a_k M^{i_k} \right\| \tag{2}$$

where for $\|\cdots\|$ we can use L_1, L_2, KL, or some other suitable notion of divergence.

3.3.2 The entropy objective

An important special case is when all $b(i)$ are equal. E.g., for each query in the well-known subtopic search data set (Section 6.1), a set of (only) relevant documents is provided. In the GCD framework, it is reasonable to want the teleported searcher to visit all these relevant documents at the same rate. A similar situation holds in the IMDB data set (Section 6.2) where the goal is to rank actors by prestige while maintaining diversity (uniform distribution) over countries. If b is uniform and KL divergence is used, (2) turns into maximizing the entropy of $\sum_{k=1}^{K} a_k M^{i_k}$.

Additional notation. Let the D dimensional probability simplex be denoted by $\mathcal{S}_D = \{p | \sum_{d=1}^{D} p_d = 1, p_d \geq 0\}$. Let the entropy $H(p) = -\sum_{d=1}^{D} p_d \log p_d$ where $p \in \mathcal{S}_D$. D can be any positive integer. The Kullback-Leibler divergence is defined as $\text{KL}(p, p') = \sum_{d=1}^{D} p_d \log \dfrac{p_d}{p'_d}$ for $p, p' \in \mathcal{S}_D$.

4. INFERENCE

There are now two questions remaining.

Inference. If α, C, b are fixed, how should the search engine design a sparse ($K = 10$ non-zeros, say) teleport $r \in \mathcal{S}_N$ such that the teleported searcher visits items at rates prescribed by b? This section focuses on this question. I.e., the output of inference is an optimal r^*, expressed as the choices of $\{i_k\}$ in (2).

Training. Suppose we are given an inference algorithm together with training data, i.e., graph instances with r^* specified. The goal of training or learning is to fit C on each graph such that the output of the inference algorithm is (close to) r^*, or at least, the inference algorithm outputs some $\tilde{r}$ achieving an objective close to that obtained with r^*. Our goal is to avoid hardwired notions of similarity between items as in much of the related work. Section 5 studies this problem.

4.1 Hardness

Proposition 1. *Even with uniform profile a, maximizing the entropy of $\sum_{k=1}^{K} a_k M^{i_k}$ is NP-hard.*

Proof. Given an exact-cover-by-3-sets (X3C [20]) instance with a universe $[T] = \{1, \ldots, T\}$ of elements (3 divides T exactly) and sets indexed $n \in [N]$, the decision version of X3C asks if there exist $K = T/3$ sets whose union is $[T]$, and each element is covered exactly once. We prepare our matrix M which will be of size $T \times N$. Each column corresponds to a set; rows corresponding to elements in the set are set to $1/3$, rest 0s. Suppose an X3C exists. This can be used to produce an r with fill $T/3$, each element being $3/T$, such that Mr is a column vector of T elements, each equal to $1/T$. The entropy is $\log T$. Suppose an X3C does not exist. Then any choice of K columns of M will leave at least one element uncovered. The form of the objective is $-\sum_i (1/T) \log p_i$, where at most $T-1$ p_i values can be positive. So the best bet is to make them all $1/(T-1)$, for an entropy of $\frac{T-1}{T} \log(T-1)$ which is strictly less than $\log T$. □

Proposition 2. *With uniform profile a and using L_2 divergence, solving (2) for arbitrary b is NP-hard.*

Proof. This follows from the hardness of finding sparse solutions to linear least-square problems [20]. □

4.2 Heuristics

Given the hardness results, it is reasonable to try a greedy search. We collect K columns from the N columns of PPV matrix M from the input set successively, in each step choosing that vector which minimizes the divergence or maximizes the entropy. The procedure is shown in Figure 1.

```
input: PPV matrix $M$; $a_1 \geq \cdots \geq a_K > 0$; reference $b$
output: sequence $S$ of $K$ columns chosen from $M$
initialize $S = \varnothing$
for $k = 1, 2, \ldots, K$ do
    for each $i \in [1, N] \setminus S$ do
        tentatively include column $i$ in $S$
        compute $\psi = \frac{\sum_{s=1}^{|S|} a_s M^s}{\sum_{s=1}^{|S|} a_s}$
        record $\|\psi - b\|$
    choose best column and commit inclusion in $S$
```

Figure 1: Greedy teleport selection.

One way to estimate the gap between the greedy solution and the optimal is to bound the optimal using a relaxed integer program. Let $z_{ki} \in \{0, 1\}$ be decision variables indicating if the ith item (column) is placed at rank k, where $i = 1, \ldots, N$ and $k = 1, \ldots, K$. The basic constraints on z_{ki} are

$$\forall k : \quad \textstyle\sum_i z_{ki} = 1 \tag{3}$$

(exactly one item in each rank, assuming $n \geq k$), and

$$\forall i : \quad \textstyle\sum_k z_{ki} \leq 1 \tag{4}$$

(each item goes to at most one rank among $1, \ldots, K$). Once z_{ki} are relaxed to $[0, 1]$, the generic divergence or entropy objectives lead to a convex optimization that can be efficiently executed.

5. LEARNING GRAPH CONDUCTANCE

In this section we discuss the design of our edge weights, and how these are learnt from data. Some salient properties of our approach are summarized below.

- Unlike prior work, we do not work with a single graph. We overlay *one graph per feature* to define edge conductance.
- The per-feature graphs are combined through a learning process.

Wherever associative graph models have been used to enhance diversity, edge conductance $C(j|i)$ have been hardwired and fixed by design. A common design has been to make $C(j|i)$ directly related to $\text{sim}(i, j)$. If $\text{sim}(i, j) \in \mathbb{R}_+$, as is the case in TFIDF cosine or Jaccard similarity,

$$C(j, i) = \frac{\text{sim}(i, j)}{\sum_{j'} \text{sim}(i, j')}. \tag{5}$$

has been frequently used. There is only one kind of edge conductance.

5.1 Edge features

Given nodes i and j, there may be multiple notions of symmetric similarity or asymmetric coverage between them. Each such notion is said to be a *feature*, indexed by $f \in [1, F]$. We will end this subsection with examples of features that we use. For each feature f, we get a conductance matrix C_f.

Now there are (at least) two ways to combine information from the per-feature conductance matrices into a single one. The first way is to use a convex combination on C_f, i.e., let $C = \sum_f \lambda_f C_f$, where $\lambda_f \geq 0$ and $\sum_f \lambda_f = 1$, i.e., $\lambda \in \mathcal{S}_F$. The resulting matrix C is also stochastic, so now we can use $M = (1 - \alpha)(\mathbb{I} - \alpha C)^{-1}$ as usual. This creates a mixture of Markov walks and then finds the corresponding PPV matrix M, which sounds more natural. However, observe that the parameters to be learnt, λ, get involved in M as a matrix inverse, which makes the optimization over λ extremely complicated. Guided by preliminary experiments, we pursue an alternate parametrization:

$$M_f = (1 - \alpha)(\mathbb{I} - \alpha C_f)^{-1}, \quad M(\lambda) = \sum_f \lambda_f M_f. \tag{6}$$

This parametrization simplifies the training procedure and is the focus of the study in the enusing paragraphs.

5.1.1 *Conventional symmetric similarity features*

Most Markov walk based diversity algorithms [32, 19] start with a symmetric edge weight based on some measure of similarity between the edge endpoints i, j. Specifically, they use cosine similarity between documents or sentences in some vector space. Other variations like (weighted) Jaccard may also be used. The raw edge weight is divided by the total outbound edge weight from a node to get outgoing transition probabilities, as shown in (5).

5.1.2 *Asymmetric coverage features (MinSim)*

Consider subtopic retrieval and a document i that is a concatenation of documents j_1 and j_2. Then i is a good center, and, to bring this out, i should have high probability transitions to j_1, j_2. The converse need not, and perhaps should not hold.

This intuition (also see [30]) led us to the "MinSim" feature. Fix a word, and suppose it occurs n_i, n_j times in document i, j where there is a directed edge (i, j). Then we say that the word contributes $\min\{1, n_i/n_j\}$ to the edge (i, j). I.e., if i contains the word as many times as j, i completely covers j as far as this word is concerned. (For $n_j = 0$ MinSim is defined to be 0.)

We can now aggregate signals from different words in various ways. We did this by taking a simple average or a weighted averages of the MinSim scores, the weights being inverse document frequency (IDF) [24].

5.1.3 *Projecting out query information*

The items we are (re)ranking for diversity are all assumed to be fairly relevant to the query. In case of subtopic retrieval, we expect that all documents in the relevant set contains some or most of the query words. One may argue that, in computing edge features, we should ignore query words. This is easy because we allow any number of potentially redundant edge features. I.e., we can compute edge features including or excluding query words, and retain both sets of features.

5.1.4 *Encoding other application signals*

Earlier work on extractive document summarization [32, 19] have observed and exploited that good summary sentences tend to come disproportionately from early parts of documents. They biased their teleport r by choosing $r(k) \propto k^{-0.25}$ where k is the rank of the sentence in a document (0.25 was found by hand-tuning). In our case, we can simply devise an additional edge feature and corresponding conductance matrix (details in Section 6.3).

5.2 Learning feature weights

During training, each instance consists of a graph skeleton, together with M_f for each feature f. For the subtopic retrieval task, an instance corresponds to a query. For the summarization task, an instance is one or more related documents that need to be summarized together. For each instance, we are also given one or more optimal, sparse teleport vectors r^*. The job of the learner is to fit a suitable λ to combine the M_fs. As a simple baseline we will start with all λ_f equal, and compare it with more elaborate algorithms.

The input data to the learner is $\{(q, M_{q,f}, r^*_q)\}$ where q is a query or task unit, M_{qf} the PPV matrix for query q and feature f, and r^*_q is/are one (or more near-) perfect teleports. For the qth query $M_q(\lambda) = \sum_f \lambda_f M_{qf}$ is a matrix and r^*_q be a given vector vector such that

$$p^q_f = M_{qf} r_q \quad \text{and} \quad p^{q*}_f = M_{qf} r^*_q, \quad \text{where} \quad p^q_f, p^{q*}_f \in \mathcal{S}_N.$$

q is omitted when fixed or clear from context. We want to select λ so that, for each query, the entropy corresponding to r^* to exceed the entropy of any other (non-optimal) teleport r. I.e., we want $H(M_q r^*_q) \geq H(M_q r)$.

5.2.1 *Entropy maximization (MaxEnt) heuristic*

Ensuring $H(Mr^*) \geq H(Mr)$ is nontrivial. As an early heuristic, we could just try to maximize the lhs. Consider the quantity

$$H(Mr^*) = H\left(\sum_f \lambda_f M_f r^*\right) = H\left(\sum_f \lambda_f p^*_f\right),$$

say, where p^*_f are *known* multinomial vectors. Because H is concave, this is a benign optimization, and can be efficiently solved. We call this the MaxEnt trainer.

5.2.2 *Exponentiated gradient (EG) approach*

We are also given an inference procedure which, given a query and the weights λ (6), outputs $\hat{r}_q$, a good if not ideal teleport. Unfortunately, the inference problem is NP hard, and can only be solved approximately. At this point it is unclear how to choose a λ to help the inference algorithm return a *good* r, given a new test query. In this section we propose a loss function and an associated subgradient based online procedure for learning λ.

5.2.2.1 *Loss function.*

In Section 4 we discussed a greedy inference algorithm which computes r with large $H(M_q(\lambda)r)$. However, the ground truth r^* may not equal r as obtained from the inference procedure.

If the inference algorithm for a fixed λ gives us r_q such that $H(M(\lambda)r_q) \geq H(M(\lambda)r^*_q)$, then clearly we need to try to improve λ. This immediately motivates a loss

$$\sum_q \Delta(r_q, r^*_q;\ \lambda) = \sum_q \left[H(M(\lambda)r_q) - H(M(\lambda)r^*_q)\right]_+ \qquad (7)$$

where $[c]_+ = \max\{0, c\}$. This loss is minimum if $H(M(\lambda)r_q) \leq H(M(\lambda)r^*_q)$ for all q. The problem of minimizing this loss as a function of λ is not easy as the loss is non-convex. In the sequel we derive a convex upper bound and derive a *prox* function based algorithm [7] for solving the problem in an online setup.

5.2.2.2 *Optimizing the loss.*

Let us rewrite the argument in the *max* function as a *difference* of two convex functions, specifically note that $\Delta(r_q, r^*_q;\ \lambda) = \max\left\{0, -G_q(\lambda) + G^*_q(\lambda)\right\}$ where $G_q(\lambda) = -H(Mr_q)$ and $G^*_q(\lambda) = -H(Mr^*_q)$. Because H is concave, G is convex in λ. Let $p(\lambda) = \sum_f \lambda_f p^q_f$ and $p^*(\lambda) = \sum_f \lambda_f p^{q*}_f$. We approximate G by its global under-estimator:

$$G_q(\lambda) \geq \hat{G}_q(\lambda) = -H\left(\sum_{i=1}^{d} \lambda^0_i p^t_i\right) + u(\lambda^0)^\top(\lambda - \lambda^0),$$

where $u_j(\lambda) = \sum_j (1+\log p_j(\lambda)) p^t_{ij}$. This motivates a convex loss function which upperbounds Δ. Specifically,

$$\Delta(r_q, r^*_q, \lambda) \leq l_q(\lambda)\left(= \max\left(0, G^*_q(\lambda) - \hat{G}_q(\lambda)\right)\right).$$

One can minimize l_q as a function of λ. Though the problem is convex, one cannot invoke gradient based procedure for this purpose as l_q is not differentiable. We exploit the fact that l_q is sub-differentiable and propose a subgradient based algorithm suited for the online setting. We use the subgradient

$$g_q(\lambda) = \begin{cases} 0, & \hat{G}_q(\lambda) \geq G^*_q(\lambda) \\ \nabla_\lambda G^*_q(\lambda) - u(\lambda^0), & \text{otherwise.} \end{cases}$$

Here $g_q(\lambda)_i = \sum_{j=1}^n (1 + \log p^*_j(\lambda)) p^{q*}_{ij}$. This immediately implies that

$$l_q(\lambda) - l_q(\lambda') \leq (\lambda - \lambda')^\top g_q(\lambda) \qquad (8)$$

Consider the online setup where queries are processed one after another. Let there exist

$$\lambda^* = \arg\min_{\lambda \in \mathcal{S}_F} L_T(\lambda)\left(= \frac{1}{T}\sum_{q=1}^{T} l_q(\lambda)\right).$$

We define the *regret* R as $R(\lambda_1, \ldots, \lambda_T) = L(\lambda_1, \ldots, \lambda_T) - L_T(\lambda^*)$ where $L(\lambda_1, \ldots, \lambda_T) = \frac{1}{T}\sum_{q=1}^T l_q(\lambda_q)$.

Proposition 3. *Let there exist m such that*

$$m \geq \|g_q(\lambda)\|_\infty \quad \forall \lambda \in \mathcal{S}_F \qquad (9)$$

Then the iteration

$$\lambda_{t+1} = \arg\min_{\lambda \in S_d} \mathrm{KL}(\lambda, \lambda_t) + \eta(\lambda - \lambda_t)^\top \nabla_{\lambda_t} l_t(\lambda_t) \qquad (10)$$

ensures that $R(\lambda_1, \ldots, \lambda_T) \leq 2m\sqrt{\frac{\log F}{T}}$ provided $\eta = \sqrt{\frac{\log F}{m^2 T}}$.

Proof. The proof follows Duchi *et al.* [7]. As a consequence of (8), it is immediate to see that

$$R = \frac{1}{T}\sum_{t=1}^{T}(l_t(\lambda_t) - l_t(\lambda^*)) \leq \frac{1}{T}\sum_{t=1}^{T}(\lambda_t - \lambda^*)^\top g_t(\lambda). \qquad (11)$$

The goal would then be to upperbound the rhs. To this end see that the iteration (10) leads to updates of the form $\lambda_{(q+1)i} = \frac{\lambda_{qi}}{Z}\exp\left(-\eta g_q(\lambda)_i\right)$ where Z is chosen so that lhs sums to 1 over i. For such a choice of λ_{t+1} one can prove

$$\mathrm{KL}(\lambda^*, \lambda_t) - \mathrm{KL}(\lambda^*, \lambda_{t+1}) \geq \eta(\lambda_t - \lambda^*)^\top g_t - \eta^2 m^2 \qquad (12)$$

See that the LHS simplifies to

$$= -\log\left(\sum_k \lambda_{tk}\exp\left(-\eta\left(\nabla l_t(\lambda_t)\right)_k\right)\right) - \eta{\lambda^*}^\top \nabla l_t(\lambda_t).$$

Exploiting $e^x \leq 1 + x + x^2$ for $0 \leq x \leq 1$, and assuming that $\eta|g_{qi}| \leq 1$,

$$\log\left[\sum_k \lambda_{qk}\exp\left(-\eta g_{qk}\right)\right] \leq \log\left[1 - \eta\lambda^\top g_q + \eta^2\sum_{i=1}^{n}\lambda_{qi}g^2_{qi}\right]$$

$$\leq \log(1 - \eta\lambda^\top g_t + m^2\eta^2) \leq -\eta\lambda^\top g_t + \eta^2 m^2.$$

The last two inequalities follow because log is monotonically increasing and is upperbounded by the identity function which proves the claim.

Using (12) and summing over all t we get

$$\sum_{t=1}^{T}(\lambda_t - \lambda^*)^\top g_t \leq \tfrac{1}{\eta}\left(\mathrm{KL}(\lambda^*, \lambda_1) - \mathrm{KL}(\lambda^*, \lambda_{t+1})\right) + \eta m^2.$$

Dropping the second term in RHS we obtain the inequality

$$\sum_{t=1}^{T}(\lambda_t - \lambda^*)^\top g_t \leq \frac{1}{\eta}\mathrm{KL}(\lambda^*, \lambda_1) + \eta T m^2$$

Taking $\lambda_1 = \vec{1}/F$, where $\vec{1}$ is a vector of all 1s, yields $\mathrm{KL}(\lambda, \lambda_1) \leq \log F$. Finally the bound is proved by noting that the minimum of $\frac{a}{\eta} + b\eta$ is obtained at $\eta = \sqrt{\frac{a}{b}}$ and the optimum value is $2\sqrt{ab}$. □

The learning algorithm tries to minimize $\Delta(r_q, r_q^*; \lambda)$ by minimizing a convex upper-bound in an online setup. The minimization is not easy as it yields a non-differentiable objective but the problem is finessed by a *prox* function approach [7].

6. EXPERIMENTS

We compared GCD with most of the recent coverage/novelty [4, 29, 28] and Markov walk [32, 19] based diversity algorithms. We explored three application areas with public data sets, reported in separate subsections next.

6.1 Diverse subtopic retrieval

In the TREC 6–8 interactive tracks, each query is associated with a set of subtopics and a set of relevant documents, and each document is marked with the set of subtopics for which it is relevant. This data set has been used earlier by Zhai *et al.* [29] and Yue *et al.* [28]. The subtopics and their coverage is used to compute diversity-cognizant recall and precision scores for retrieval algorithms, but the algorithms are not supposed to know subtopic coverage information.

We used the following edge features:

- Average MinSim coverage of one document by another, weighted by word IDF in the corpus of relevant documents for each query.
- Symmetric cosine and Jaccard similarity between documents.

Each query is associated with only relevant (to at least one subtopic) documents. Thus, as in Yue *et al.* [28], the relevance issue is thereby fixed for all algorithms, and only their effect on diversity is studied. We used a uniform b. For training, r^* was obtained using greedy set cover given the subtopic/s covered by each relevant document.

For evaluation, we used standard S-recall and S-precision originally proposed by Zhai *et al.* [29] for such data. Let $\mathrm{SubTopics}(d)$ be the subtopics for which document d is relevant, and let a query have τ subtopics. The S-recall at rank K is $\frac{1}{\tau}\left|\bigcup_{k=1}^{K} \mathrm{SubTopics}(d_k)\right|$. Let $\mathrm{MinRank}(\pi, r)$ be the smallest rank at which S-recall of r is achieved by a ranking π. Then S-precision at recall r is $\frac{\mathrm{MinRank}(\pi^*, r)}{\mathrm{MinRank}(\pi, r)} \in [0, 1]$, where π^* is the optimal order. Let $\mathrm{SubTopicsUpto}(k) = \bigcup_{\kappa=1}^{k} \mathrm{SubTopics}(d_\kappa)$ and $\mathrm{NewTopics}(d_k) = \mathrm{SubTopics}(d_k) \setminus \mathrm{SubTopicsUpto}(k-1)$. Then the S-MAP (mean average precision) at rank K is defined as $\sum_{k=1}^{K} \frac{\mathrm{NewTopics}(d_k)}{\tau\, k}$, a direct analog of standard MAP.

6.1.1 *Accuracy of diversification*

As a summary view, Figure 2 shows S-precision vs. S-recall for most of the recent algorithms. GCD stands out as dominating other algorithms over almost all recall levels. DivRank is excellent at small recall, and, given its simplicity, MMR is surprisingly good at larger recall.

Figure 3 shows subtopic-aware MAP (mean average precision) or S-MAP against rank of responses. Our GCD approach is clearly superior at all ranks. SVMdiv and DivRank are the runners-up.

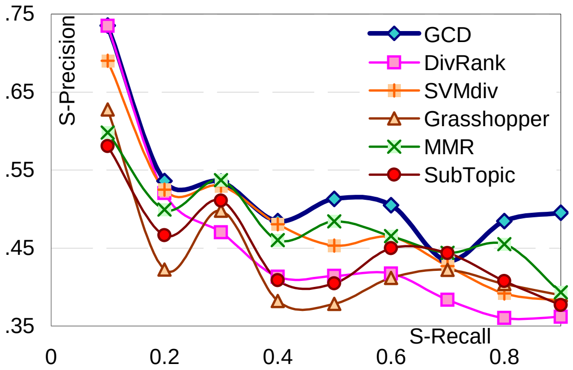

Figure 2: S-precision vs. S-recall, subtopic retrieval.

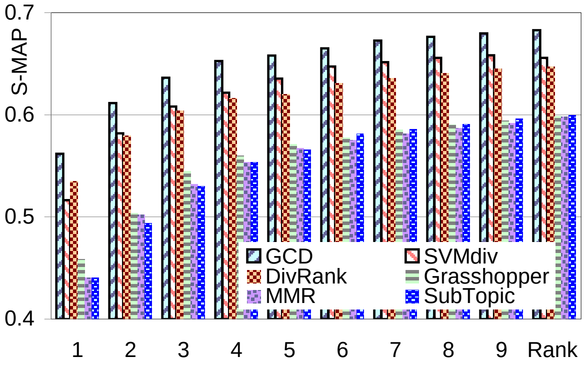

Figure 3: Subtopic-aware MAP (S-MAP).

6.1.2 *Inference quality*

Greedy search (Section 4.2) was used throughout. We did spot-checks on 17 queries, comparing relaxed integer program (Section 4.2) entropy (upper bound) with greedy entropy; average upper bound ÷ greedy was **1.0089**; it was less than 1.008 for 16 queries.

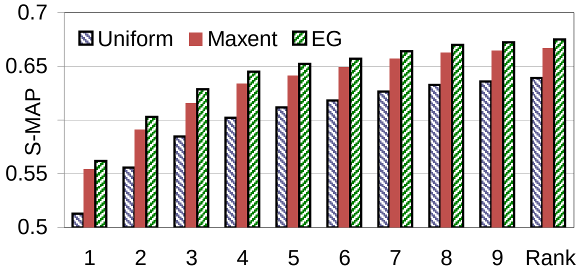

Figure 4: Effect of training λ.

6.1.3 *Effect of training*

In Figure 4 we show test S-MAP at ranks 1–10 after various forms of training λ. The baseline is no training, with all λ_f equal, i.e., giving equal importance to all M_f matrices. The second set of bars correspond to the MaxEnt trainer described in Section 5.2.1. The final set of bars corresponds to the exponentiated gradient (EG) trainer described in Section 5.2.2. MaxEnt training shows a substantial benefit beyond using a uniform λ, and EG training further improves on that. The typical training time per instance of GCD is less than 10 seconds, while for SVMdiv it is over 3 hours.

6.1.4 *Ablation study*

In GCD we changed both the edge representation and the algorithm. Compared to DivRank and Grasshopper,

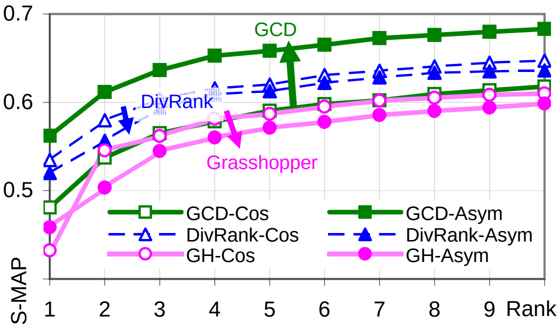

Figure 5: Ablation study.

how much of the accuracy gains shown by GCD are because of a different graph, and how much of it is because of a different, trainable algorithm? Figure 5 shows that moving from symmetric cosine to asymmetric MinSim greatly helps GCD, while it hurts DivRank and Grasshopper. In fact, GCD with cosine is worse than DivRank. This highlights the unified nature of our formulation.

6.2 Ranking in social networks

Grasshopper [32] introduced a diversity task quite distinct from subtopic retrieval and document summarization. It involves the IMDB database, represented as a graph with actors as nodes. Associated with each actor is a set of movies where s/he worked. Edges between actors nodes can be designed in various ways, depending on symmetric overlap (cosine, Jaccard) or asymmetric coverage (MinSim) in terms of movies associated with the endpoint nodes. Unseen by the algorithms, each actor is also associated with a country. The data involves 3452 actors, 1027 movies, and 47 countries.

Each algorithm has to rank the actors by some notion of network prestige. Since high-prestige actors work in many movies, we expect that, as we collect more actors down the ranked list, we will rapidly increase the number of associated movies. If the ranking of actors is also diverse, we would expect to collect countries rapidly as well.

Figure 6 shows the number of distinct countries and movies associated with a prefix of top-ranked actors from the lists returns by various algorithms, where parameters were chosen to maximize the performance of each algorithm. DivRank works well, but the diversity of GCD dominates others wrt both movies and countries.

6.3 Document summarization

Earlier diversity algorithms [32, 19] were evaluated on sentence-level document summarization. The goal is to return a set of sentences that have low redundancy among themselves, and "covers" well the rest of the sentences in the document/s being summarized. Representing sentences as nodes and symmetric similarity or asymmetric coverage as edges between sentence pairs makes it natural to apply Grasshopper, DivRank and GCD.

Prior work recognized that earlier sentences in documents are more likely to be included in good summaries, and encoded this in their teleport vectors. In GCD, since the teleport is our *output*, we need a different device. Within a document, sentence number i links to all sentences numbered $j > i$, and the last sentence loops back to the first one. This corresponds to the realistic reader/surfer model of reading a document starting at a given sentence, getting bored with some probability at each subsequent sentence and teleporting away.

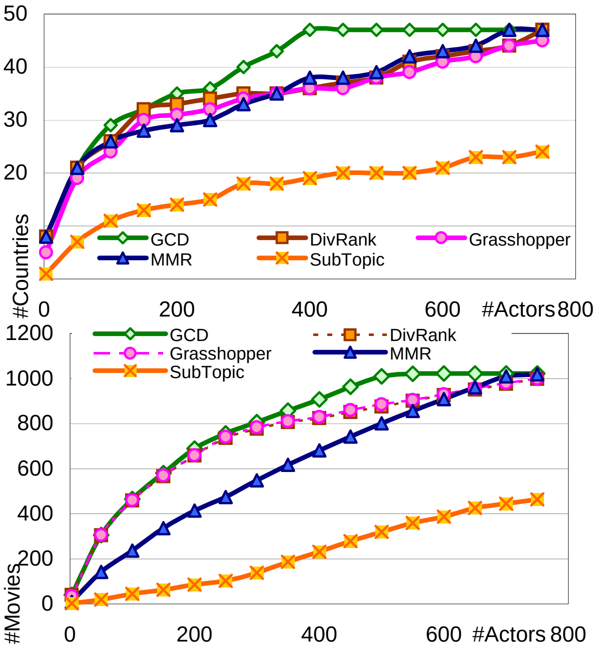

Figure 6: Simultaneous diverse country and movie coverage as a function of number of top actors.

The DUC 2004 data set includes human-written, *non-extractive* summaries with associated ROUGE-1 scores [32]. We exhaustively searched the documents for sets of sentences that have ROUGE-1 scores close to the human summaries, and use these sets to define r^* for training. We used the 50 labeled instances as defined by Task-2 of DUC 2004, randomly split into 30 training and 20 test instances.

Algorithm	Train	Test
MMR [4]	.324	.32
SubTopic [29]	.32	.323
Grasshopper [32]	.341	.33
DivRank [19]	.353	.345
GCD	.365	**.369**
Submodular [16]	.389	**.373**
Optimal	.421	.407

Figure 7: Document summarization accuracy.

Figure 7 shows ROUGE-1 scores for different algorithms. GCD fares better than MMR, SubTopic, Grasshopper and DivRank. Submodular beats GCD by only **0.004** in test data. Here, GCD did not use information about sentence length and summary budget (in words or bytes), whereas Submodular did. Leveling the playing field and comparing them again is left for future work.

6.4 Typical training and inference time

Recent work on diversity focuses more on formulation and quality than speed. In particular, evaluating M_f through matrix inversion can be expensive. On the TREC subtopic data set, SVMDiv takes over three hours to train. As a ballpark estimate, for executing the average query, SVMDiv takes 1s, GCD and Zhai *et al.*'s subtopic algorithms take 2s each, Grasshopper takes 4s, and DivRank takes 3s. For IMDB, query times compare as Zhai $<$ GCD $<$ DivRank $\ll$ Grasshopper. Thus, although none of these algorithms are well suited for large-scale real-time deployment, GCD is quite competitive wrt prior art.

7. CONCLUSION

The starting point of our work was to recognize that diversity algorithms [32, 19] modeled around PageRank had no plausible generative or phenomenological explanation. Meanwhile, SVMDIV [28] can be trained to recognize topic coverage, and NETRANK [1] can train graph conductances. The key difference between NETRANK and this work is that here we are after *dissociative* selection, which is achieved by the graph center search. In this process, we gave an alternative, sound theoretical foundation of diversity around associative graph models. Giving inference guarantees and extending to clickthrough data are natural directions of future work.

Acknowledgment. Thanks to Andrew Goldberg and Qiaozhu Mei for help with data sets and Yisong Yue for help with SVMDIV. Thanks to the reviewers for helping us improve the presentation.

8. REFERENCES

[1] A. Agarwal and S. Chakrabarti. Learning random walks to rank nodes in graphs. In *ICML*, 2007.

[2] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying search results. In *WSDM Conference*, pages 5–14, 2009.

[3] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[4] J. Carbonell and J. Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *SIGIR Conference*, pages 335–336, 1998.

[5] H. Chen and D. Karger. Less is more: probabilistic models for retrieving fewer relevant documents. In *SIGIR Conference*, pages 429–436, 2006.

[6] A. Das Sarma, S. Gollapudi, and S. Ieong. Bypass rates: reducing query abandonment using negative inferences. In *SIGKDD Conference*, pages 177–185, 2008.

[7] J. Duchi, S. Shalev-Shwartz, Y. Singer, and A. Tewari. Composite objective mirror descent. In *COLT*, pages 14–26, 2010.

[8] K. El-Arini, G. Veda, D. Shahaf, and C. Guestrin. Turning down the noise in the blogosphere. In *SIGKDD Conference*, pages 289–298. ACM, 2009.

[9] A. Frieze, J. Vera, and S. Chakrabarti. The influence of search engines on preferential attachment. *Internet Mathematics*, 3(3):361–381, 2006–2007.

[10] L. A. Granka, T. Joachims, and G. Gay. Eye-tracking analysis of user behavior in www search. In *SIGIR Conference*, pages 478–479, 2004.

[11] S. Guo and S. Sanner. Probabilistic latent maximal marginal relevance. In *SIGIR Conference*, pages 833–834, 2010. Poster.

[12] G. Jeh and J. Widom. Scaling personalized web search. In *WWW Conference*, pages 271–279, 2003.

[13] D. Kempe, J. M. Kleinberg, and É. Tardos. Influential nodes in a diffusion model for social networks. In *ICALP*, pages 1127–1138, 2005.

[14] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[15] L. Li, K. Zhou, G.-R. Xue, H. Zha, and Y. Yu. Enhancing diversity, coverage and balance for summarization through structure learning. In *WWW Conference*, pages 71–80, Apr. 2009.

[16] H. Lin and J. Bilmes. Multi-document summarization via budgeted maximization of submodular functions. In *HLT Conference*, pages 912–920, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[17] H. Lin, J. Bilmes, and S. Xie. Graph-based submodular selection for extractive summarization. In *Automatic Speech Recognition and Understanding Workshop*, 2009.

[18] T.-Y. Liu. Learning to rank for information retrieval. Tutorial at SIGIR, 2008.

[19] Q. Mei, J. Guo, and D. Radev. DivRank: the interplay of prestige and diversity in information networks. In *SIGKDD Conference*, pages 1009–1018, 2010.

[20] B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24(2):227–234, apr 1995.

[21] T. Qin, T.-Y. Liu, X.-D. Zhang, D.-S. Wang, W.-Y. Xiong, and H. Li. Learning to rank relational objects and its application to Web search. In *WWW Conference*, pages 407–416, 2008.

[22] F. Radlinski and T. Joachims. Active exploration for learning rankings from clickthrough data. In *SIGKDD Conference*, pages 570–579, 2007.

[23] F. Radlinski, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. In *ICML*, pages 784–791, 2008.

[24] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.

[25] H. Tong and C. Faloutsos. Center-piece subgraphs: Problem definition and fast solutions. In *SIGKDD Conference*, 2006.

[26] H. Tong, C. Faloutsos, and J.-Y. Pan. Fast random walk with restart and its applications. In *ICDM*, 2006.

[27] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 6(Sep):1453–1484, 2005.

[28] Y. Yue and T. Joachims. Predicting diverse subsets using structural SVMs. In *ICML*, pages 271–278, 2008.

[29] C. X. Zhai, W. W. Cohen, and J. Lafferty. Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *SIGIR Conference*, pages 10–17, 2003.

[30] B. Zhang, H. Li, Y. Liu, L. Ji, W. Xi, W. Fan, Z. Chen, and W.-Y. Ma. Improving web search results using affinity graph. In *SIGIR Conference*, SIGIR '05, pages 504–511, New York, NY, USA, 2005. ACM.

[31] Y. Zhang, J. Callan, and T. Minka. Novelty and redundancy detection in adaptive filtering. In *SIGIR Conference*, pages 81–88, 2002.

[32] X. Zhu, A. B. Goldberg, J. Van, and G. D. Andrzejewski. Improving diversity in ranking using absorbing random walks. In *HLT-NAACL*, pages 97–104, 2007.