Regularizing Ad Hoc Retrieval Scores

Fernando Diaz
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
fdiaz@cs.umass.edu

ABSTRACT

The cluster hypothesis states: closely related documents tend to be relevant to the same request. We exploit this hypothesis directly by adjusting *ad hoc* retrieval scores from an initial retrieval so that topically related documents receive similar scores. We refer to this process as score regularization. Score regularization can be presented as an optimization problem, allowing the use of results from semi-supervised learning. We demonstrate that regularized scores consistently and significantly rank documents better than un-regularized scores, given a variety of initial retrieval algorithms. We evaluate our method on two large corpora across a substantial number of topics.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval—*clustering, retrieval models*

General Terms

Algorithms, Performance, Experimentation

Keywords

regularization, manifold learning, pseudo-relevance feedback, clustering

1. INTRODUCTION

In *ad hoc* retrieval, the clustering hypothesis states: *closely related documents tend to be relevant to the same request* [7]. Many information retrieval techniques have adopted the clustering hypothesis as a core assumption. A number of methods explicitly attempt to partition the corpus into clusters. Some examples of this approach include cluster-based retrieval, latent semantic indexing, and aspect models. Other methods build clusters on the fly in response to a query. These methods include pseudo-relevance feedback and query-dependent clustering.

CIKM'05, October 31–November 5, 2005, Bremen, Germany.

In score-based retrieval, the clustering hypothesis implies the following: *closely related documents should have similar scores, given the same request.* We propose expressing this implication as an optimization problem of balancing the score from some initial retrieval with the scores of related documents. When viewed as a process of smoothing document scores with those of related documents, this problem can be solved with methodologies from machine learning. We refer to this process as *score regularization.*

We will begin by describing the general regularization framework in Section 2. This regularization framework relies on having a data structure to encode document relatedness. In Section 3, we present a method for computing relatedness when explicit information is absent. The clustering hypothesis underlies many important information retrieval techniques. In Section 4, we reduce several well-known techniques to score regularization. We present results for regularizing *ad hoc* retrieval scores in Section 5. We conclude in Section 6 by placing our work in context of previous results in machine learning and information retrieval.

2. LOCAL SCORE REGULARIZATION

In previous work, regularization has been posed as an optimization problem [27]. We will review relevant results in the context of information retrieval. More thorough derivations can be found in referenced publications.

Let n be the number of document scores to regularize. In our case, this is always the top n documents of an initial retrieval. Given the initial scores in the length n vector, y, we would like to compute a set of regularized scores, f, also a length n vector. We have two contending objectives: score consistency with related documents and score consistency with the initial retrieval. Let $\mathcal{S}(f)$ be a cost function associated with the inter-document consistency of the scores, f; if related documents have very inconsistent scores, then the value of this function will be high. Let $\mathcal{E}(f)$ be a cost function measuring the consistency with the original scores; if documents have scores very inconsistent with their original scores, then the value of this function will be high. We use a linear combination of these objectives for our composite objective function,

$$\mathcal{Q}(f) \quad = \quad \mathcal{S}(f) + \mu\mathcal{E}(f) \tag{1}$$

where μ is a regularization parameter allowing us to control how much weight to place on inter-document smoothing versus consistency with the original score.

2.1 Measuring Inter-Document Consistency

Inter-document relatedness is represented by an $n \times n$ affinity matrix, W, where W_{ij} represents the affinity between documents i and j and $W_{ii} = 0$. At the moment, we will leave the notion of affinity abstract, allowing any number of possible measures; we will outline one way to build this matrix in Section 3. A set of scores is considered *smooth* if related documents tend to have similar scores. To this end, we define the cost function, $\mathcal{S}(f)$, which penalizes inconsistency between related documents,

$$\mathcal{S}(f) = \sum_{i,j=1}^{n} \left(\sqrt{\frac{W_{ij}}{D_{ii}}} f_i - \sqrt{\frac{W_{ij}}{D_{jj}}} f_j \right)^2 \quad (2)$$

where D is a diagonal normalizing matrix such that $D_{ii} = \sum_{j=1}^{n} W_{ij}$. This matrix, D, allows us to normalize the affinity between two documents; the diagonal elements of this matrix represent how related a document is to all other documents. We can then weight the affinity between some document d_i and d_j relative to its affinity with all other documents. Using such a normalization has been shown to have superior convergence properties than unnormalized affinities for tasks such as spectral clustering [23].

2.2 Measuring Consistency with Initial Scores

Obviously, we would like to select our regularized scores, f, such that $\mathcal{S}(f)$ is minimized. Unconstrained, however, minimizing this objective function would yield a flat set of scores. Therefore, we consider a second objective so that the regularization does not stray from the initial scores,

$$\mathcal{E}(f) = \sum_{i=1}^{n} (f_i - y_i)^2 \quad (3)$$

The regularized scores, f, minimizing this function would be completely consistent with the original scores, y; that is, if we only minimize this objective, then the solution is $f = y$.

2.3 Computing the Optimal Scores

We would like to find the optimal set of regularized scores, f^*, such that,

$$f^* = \arg\min_{f \in \Re^n} \mathcal{Q}(f) \quad (4)$$

We can compute the solution to this problem by iteratively smoothing the scores. This computation can be formulated as,

$$f^{t+1} = (1-\alpha)y + \alpha D^{-1/2} W D^{-1/2} f^t \quad (5)$$

where $\alpha = \frac{1}{1+\mu}$ is a simple function of our regularization parameter [27]. We can initialize the regularized scores such that $f^0 = y$. In the limit, the regularized scores, f^t, converge on the optimal scores, f^*. Notice here that, for a single iteration, a candidate document score is smoothed with the scores of its related neighbors weighted by the relative affinity with individual neighbors.

Alternatively, the optimal regularized scores can be formulated as the solution of matrix operations,

$$f^* = \left(I - \alpha D^{-1/2} W D^{-1/2} \right)^{-1} y \quad (6)$$

In our experiments, we will use the closed form solution in Equation 6 to compute f^*.

3. COMPUTING THE AFFINITY MATRIX

The affinity matrix, W, defines the behavior of the regularization. A poor affinity matrix will result in the smoothing of scores between unrelated documents. Oftentimes, the affinity is explicit and suitable for topical relationships. For example, hyperlinks can provide evidence that two documents share a topic. When such information is not available, affinity can be computed using any number of measures of document similarity.

The majority of results presented here use language modeling baseline systems. In order to maintain consistency between our affinity measure and our retrieval model, we will focus on distributional affinity of document language models.[1] One popular distributional affinity measure in the information retrieval community is the Kullback-Leibler divergence. However, this measure is asymmetric and has demonstrated mixed results when made symmetric. Therefore, we use the multinomial diffusion kernel [12]. This affinity measure between two distributions, θ_i and θ_j, is motivated by Fisher information metric and defined as,

$$\mathcal{K}(\theta_i, \theta_j) = (4\pi t)^{-\frac{|V|}{2}} \exp\left(-t^{-1} \arccos^2 \left(\sqrt{\theta_i} \cdot \sqrt{\theta_j} \right) \right) \quad (7)$$

where $|V|$ is the size of the vocabulary and t is a parameter controlling the decay of the affinity with respect to the arc cosine of the component-wise square root of the distributions. In information retrieval, $|V|$ can be quite large. Therefore, for practical reasons, we ignore the first term, $(4\pi t)^{-\frac{|V|}{2}}$. The diffusion kernel has been shown to be a good affinity metric for tasks such as text classification.

Although we compute the complete $n \times n$ affinity matrix, there are several reasons to consider a sparse affinity matrix instead. For example, we may be more confident about the affinity between very related documents than distant documents. In this situation, the space is often better approximated by the geodesic distances between documents; that is, using the piecewise affinity over local relationships as a measure of non-neighboring affinity rather than ambient affinity. Such a sparse representation can be reasoned about as a weighted graph capturing potential lower-dimensional structure in the data. For example, the matrix may only include the affinities to the k-nearest documents and zero otherwise. A growing body of work has demonstrated that constructing these document affinity graphs accurately captures the lower-dimensional manifold [3]. Our preliminary work confirmed that using the complete affinity matrix was not as successful as sparser representations.

We should, at this point, list a few caveats about assuming the presence of an underlying, lower-dimensional manifold. First, there is no explicit evidence that the documents from the initial retrieval lie on a lower-dimensional manifold. However, the success of cluster-based retrieval methods suggest that there probably exists some topical substructure [14, 26]. Second, the use of these manifold methods normally assumes a uniform sampling on the manifold. We know, though, that topics are neither similarly sized nor uniformly sampled in the initial retrieval. We therefore note

[1] We should note that there is no reason why we could not use cosine similarity in computing W. Preliminary experiments have shown that the retrieval power of language models combined with the well-studied cosine similarity measure leverage the strengths of both.

1.	compute $n \times n$ affinity matrix
2.	add the k nearest neighbors for each document to W
3.	$D_{ii} = \sum_{i \neq j} W_{ij}$
4.	$f^* = \left(I - \alpha D^{-1/2} W D^{-1/2}\right)^{-1} y$
n	number of document scores to regularize
y	top n initial retrieval scores
k	number of neighbors to consider
α	parameter favoring inter-document consistency
f^*	regularized scores

Figure 1: Local Score Regularization Algorithm. Inputs are n, y, k, and α. The output is the a length n vector of regularized scores, f^*.

that our performance can be further improved by addressing some of these issues in future work.

Our final score regularization algorithm is presented in Figure 1. Note that the affinity matrix computed in Step 1 is used for adding elements to W in Step 2 and does not define W itself unless $k = n$.

4. RELATIONSHIP TO OTHER MODELS

Several classic retrieval models can be posed as instances of score regularization. In this section, we will be focusing on the relationship between these models and a single iteration of score regularization. To this end, we define a general version of Equation 5. Given an initial score function, $s^0(d, Q)$, and affinity matrix, A_{d_i,d_j}, a regularized score can be computed as,

$$s^1(\mathbf{d}, Q) \quad = \quad \alpha_{\mathcal{E}} s^0(\mathbf{d}, Q) + \alpha_{\mathcal{S}} \sum_{\delta} s^0(\delta, Q) A_{\delta,\mathbf{d}} \qquad (8)$$

where the bolded $\mathbf{d}$ is our candidate document and $\alpha_{\mathcal{E}}$ and $\alpha_{\mathcal{S}}$ are our regularization weights. The index δ is some item with which we are comparing the candidate document. These could be other documents or—in some cases—clusters. We will also see circumstances where s^0 behaves one way for $\mathbf{d}$ and another way for δ.

Given this definition, Equation 5 can be writtern as,

$$s^1(\mathbf{d}, Q) \quad = \quad (1 - \alpha) y_{\mathbf{d}} + \alpha \sum_{i=1}^{n} y_i [D^{-1/2} W D^{-1/2}]_{\mathbf{d},i} \qquad (9)$$

4.1 Pseudo-Relevance Feedback

Pseudo-relevance feedback refers to the technique of building a model out of the top n documents retrieved by the original query. The system then performs a second retrieval using combination of this model and the original query.

4.1.1 Rocchio

The classic Rocchio pseudo-relevance feedback algorithm assumes some number of the top documents from an initial retrieval to be relevant. Let this set be R. We then linearly combine these document vectors with the original query vector [20]. Using normalized document and query vectors, the modified query can be computed by,

$$Q' \quad = \quad Q + \frac{\alpha_R}{|R|} \sum_{d \in R} d \qquad (10)$$

where α_R is the weight placed on the pseudo-relevant documents. We can then use this new representation to score documents by their cosine similarity to Q'. This allows us to derive the regularization version of Rocchio,

$$\begin{aligned} \cos(\mathbf{d}, Q') &= \cos\left(\mathbf{d}, Q + \frac{\alpha_R}{|R|} \sum_{d \in R} d\right) \\ &\propto \sum_{w \in V} \mathbf{d}_w \left(Q_w + \frac{\alpha_R}{|R|} \sum_{d \in R} d_w\right) \\ &= \sum_{w \in V} \mathbf{d}_w Q_w + \frac{\alpha_R}{|R|} \sum_{d \in R} \sum_{w \in V} \mathbf{d}_w d_w \\ &= \cos(\mathbf{d}, Q) + \frac{\alpha_R}{|R|} \sum_{d \in R} \cos(\mathbf{d}, d) \qquad (11) \end{aligned}$$

Notice here that the first factor in the sum is merely the original cosine similarity between the document and query. The second factor in the sum represents the similarity to the pseudo-relevant documents. In terms of Equation 8, the δ indexes over only the pseudo-relevant documents.

We can also look at a version of Rocchio where we assume a pseudo-non-relevant document set, N, usually sampled from the tail of an initial ranking. A similar derivation results in,

$$cos(\mathbf{d}, Q) \quad = \quad \cos(\mathbf{d}, Q) + \frac{\alpha_R^R}{|R|} \sum_{d \in R} \cos(\mathbf{d}, d) - \frac{\alpha_R^N}{|N|} \sum_{d \in N} \cos(\mathbf{d}, d)$$

where we now consider two weight parameters. Notice that $s(\delta, Q) = \{-1, 1\}$ behaves very differently than $s(d, Q) = \cos(d, Q)$. Effectively, the regularization only propagates scores of documents in $R \cup N$; and for these documents, the scores are 1 or -1.

4.1.2 Relevance Models

A far more interesting case arises with the language model version of pseudo-relevance feedback [13]. In this case, the original scores are used as weights for the estimated relevance model. This relevance model, $P(w|\theta_R)$, is formally constructed by interpolating the maximum likelihood query model, $P(w|\theta_Q)$, and document models, $P(w|\theta_d)$,

$$P(w|\theta_R) \quad = \quad \lambda P(w|\theta_Q) + (1 - \lambda) \left(\sum_{d \in R} \frac{P(Q|\theta_d)}{\mathcal{Z}} P(w|\theta_d)\right) \qquad (12)$$

where $\mathcal{Z} = \sum_{D \in R} P(Q|\theta_d)$. Theoretically, the summation goes over the entire collection so that R includes every document.

We can use the cross entropy between language models as a scoring measure. Defined as the dot product of language model vectors, the cross entropy is,

$$\theta_R \cdot \log \theta_{\mathbf{d}} \quad = \quad \sum_{w \in V} P(w|\theta_R) \log P(w|\theta_{\mathbf{d}}) \qquad (13)$$

where θ is a vector of term probabilities. After some algebraic manipulation,[2] we can rewrite the ranking based on

[2]Credit is due to Victor Lavrenko for suggesting this derivation.

	δ	$s^0(\mathbf{d}, Q)$	$s^0(\delta, Q)$	$A_{\delta,\mathbf{d}}$	$\alpha_{\mathcal{E}}$	$\alpha_{\mathcal{S}}$
Rocchio	top $\|R\|$	$\cos(\mathbf{d}, Q)$	1	$\cos(\mathbf{d}, d)$	1	$\frac{a}{\|R\|}$
Relevance Model	top $\|R\|$	$\theta_Q \cdot \log \theta_\mathbf{d}$	$\frac{\theta_Q \cdot \log \theta_d}{\mathcal{Z}/\|Q\|}$	$\theta_d \cdot \log \theta_\mathbf{d}$	$1-\lambda$	λ
Cluster-based Retrieval	C	$\theta_Q \cdot \log \theta_\mathbf{d}$	$\theta_Q \cdot \log \theta_c$	$P(\mathbf{d}\|c)$	$(1-\lambda)\alpha_c$	λ
Local Score Regularization	top $\|R\|$	$P(Q\|\theta_\mathbf{d})$	$P(Q\|\theta_d)$	$(D^{-1/2}WD^{-1/2})_{\mathbf{d},d}$	α	$1-\alpha$

Table 1: Generalized Regularization: many classic algorithms in information retrieval can be re-interpreted as instances of a single iteration of our regularization algorithm. The general regularization form is presented in Equation 8.

Equation 12 as,

$$\begin{aligned}\theta_R \cdot \log \theta_\mathbf{d} &= \lambda \left(\theta_Q \cdot \log \theta_\mathbf{d}\right) \\ &+ (1-\lambda) \sum_{d \in R} \left(\frac{\theta_Q \cdot \log \theta_d}{\mathcal{Z}/|Q|} \right) (\theta_d \cdot \log \theta_\mathbf{d})\end{aligned} \tag{14}$$

using the fact that $P(Q|\theta_d) = |Q|(\theta_Q \cdot \log \theta_d)$. Whereas the Rocchio method assume the top n to have score 1.0, this method uses the normalized document score $\frac{\theta_Q \cdot \log \theta_d}{\mathcal{Z}/|Q|}$.

4.2 Cluster-based Retrieval

As mentioned earlier, several techniques attempt to explicitly cluster documents and use this information in retrieval. We will investigate a recently proposed language model version of cluster-based retrieval [9]. We can extend derivations from this work to demonstrate that cluster-based retrieval is an instance of regularization,

$$\begin{aligned}P(Q|\mathbf{d}) &= \sum_{c \in C} P(Q|\mathbf{d}, c) P(c|\mathbf{d}) \\ &= \sum_{c \in C} (\lambda P(Q|\theta_\mathbf{d}) + (1-\lambda) P(Q|\theta_c)) P(c|\mathbf{d}) \\ &= \lambda P(Q|\theta_\mathbf{d}) + (1-\lambda)\alpha_c \sum_{c \in C} P(Q|\theta_c) P(\mathbf{d}|c) \\ &\propto \lambda(\theta_Q \cdot \log \theta_\mathbf{d}) \\ &+ (1-\lambda)\alpha_c \sum_{c \in C} (\theta_Q \cdot \log \theta_c) P(\mathbf{d}|c)\end{aligned} \tag{15}$$

where C is the set of clusters and $\alpha_c = \frac{P(c)}{P(\mathbf{d})}$ is constant for all documents. Notice the similarity of Equations 14 and 15. The first factor in the sum is the same in both. The second factor corresponds to our smoothing process. Instead of smoothing against the scores of all documents in the collection, we weight against the scores of clusters. The value of $P(\mathbf{d}|c)$ can be computed in various ways. Previous work has used an exponential function of the negative Kullback-Leibler divergence [9]. This results in a non-symmetric measure whose behavior is very similar to our diffusion kernel.

We present the results of these reductions in Table 1. It is worthwhile to make some observations. First, only Rocchio and local score regularization use symmetric affinity measures. Symmetry lets us make certain assumptions about the affinity matrix, allowing us to use the results presented in Section 2. Second, some algorithms handle the normalization of $s^0(\mathbf{d}, Q)$ and $s^0(\delta, Q)$ differently. Normalization presumably puts scores on similar scales. Both relevance models and cluster-based retrieval use unnormalized scores for the $s^0(\mathbf{d}, Q)$; only relevance models normalizes $s^0(\delta, Q)$. In our experiments, we normalize both $s^0(\mathbf{d}, Q)$ and $s^0(\delta, Q)$ in a consistent manner. Third, only local score regularization normalizes the affinity. As mentioned earlier, normalizing the affinity has nice theoretical properties. In our experiments, we found affinity normalization to be critical for good performance.

5. REGULARIZING AD HOC RETRIEVAL SCORES

5.1 Evaluation

5.1.1 Topics

We performed all experiments on two data sets. The first data set, trec12, consists of the 150 TREC *ad hoc* topics 51-200. We used all collections on Tipster disks 1 and 2 [6]. The second data set, robust, consists of the 250 TREC 2004 Robust topics [24]. These topics are considered to be difficult and have been constructed to focus on topics which systems usually perform poorly on. For both data sets, we use only the topic title field. We indexed collections using the Lemur toolkit, the Rainbow stop word list, and Krovetz stemming [1, 16, 8].

5.1.2 Training

We performed exhaustive grid search to train our two free parameters: α and t. The regularization parameter was swept over values $\alpha = [0.1, 0.9]$ with a step size of 0.1. The kernel spread parameter was swept over values $t^{-1} = [0.1, 0.9]$ with a step size of 0.1. We considered 10 nearest neighbors in accordance with previous document classification results.

We selected parameters to optimize mean average precision. We present mean average precision results as well as interpolated precision at the standard 11 recall points.

5.1.3 Cross Validation

We performed 10-fold cross-validation by randomly partitioning the topics described in Section 5.1.1. For each partition, i, the algorithm is trained on all but that partition and is evaluated using that partition, i. For example, if the training phase considers the topics and judgments in partitions 1-9, then the testing phase uses the optimal parameters for partitions 1-9 to perform retrieval using the topics in partition 10. Performing this procedure for each of the ten partitions results in 150 ranked lists for trec12 or 250 for robust. Evaluation was performed using the concatenation of these ranked lists.

5.1.4 Pool Size

We believe that the top of the initial retrieval tends to be more topically consistent than the full ranked list. This

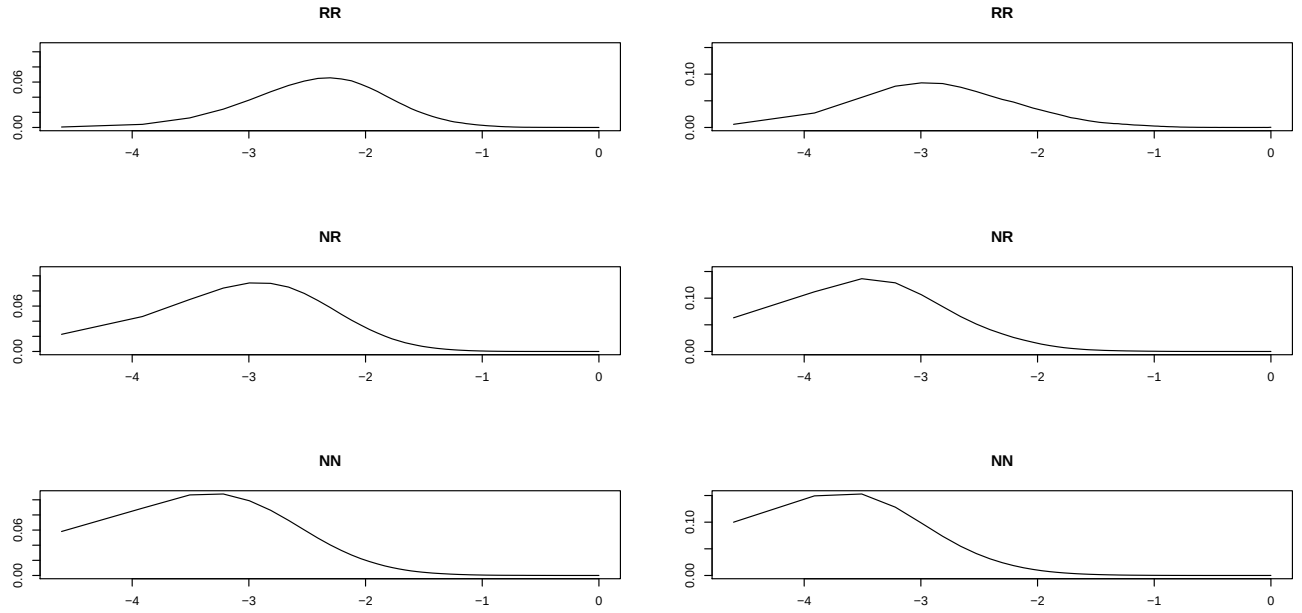

Figure 2: Distribution of log-cosine scores between relevant documents (RR), non-relevant and relevant documents (NR), and non-revelant documents (NN) in an initial retrieval of 1000 documents (left: trec12, right: robust). Relevant documents tend to be closer to each other than non-relevant documents. This suggests that the relevant document exist in some core of the initial retrieval. On the other hand, non-relevant documents are spread farther apart indicating more of a diffuse, topic-less behavior.

belief follows from an investigation of the distances between known relevant and known non-relevant documents in the initial retrieval. Figure 2 displays the distribution of log-cosine similarities between relevant documents and non-relevant documents; a similar graph is presented in the original clustering hypothesis work. As can be seen, relevant documents—those near the top of the ranked list—tend to be more similar than non-relevant documents. This indicates that the relevant documents exist in some small core of the initial retrieval and the non-relevant documents spread out from this core without relationship to each other. We therefore present results for regularizing the top 100, 250, 500, and 1000 results. This presentation also lets us note the improvement we can achieve if we do not have the resources to compute the full 1000×1000 affinity matrix.

5.2 Baseline Scores

Our experiments considered two retrieval systems: language models and Okapi.

5.2.1 Language Modeling Scores

As baselines we use query-likelihood retrieval [4] and relevance models [13]. Both of these algorithms are implemented in the Lemur language modeling toolkit [1]. The query-likelihood is a standard language model scoring technique and ranks documents according to the probability of the document having generated the query,

$$P(Q|\theta_d) \quad = \quad \prod_{q \in Q} P(q|\theta_d)^{\#(q,Q)} \tag{16}$$

where θ_d is the document language model. We use the log of this score in our experiments. As mentioned earlier, relevance models are the language model equivalent of pseudo-relevance feedback and use an initial ranking generated by Equation 16. Using the classic formulation, we first estimate a relevance model,

$$P(w|\theta_R) \quad = \quad \sum_{d \in R} P(w|D) \frac{P(Q|\theta_d)}{\sum_{d' \in R} P(Q|\theta_{d'})} \tag{17}$$

where R is the set of top 50 documents. We then rank the documents according to the Kullback-Leibler divergence between documents and the relevance model,

$$D_{KL}(P(\cdot|\theta_R)||P(\cdot|\theta_d)) \quad = \quad \sum_{w \in V} P(w|\theta_R) \log \frac{P(w|\theta_R)}{P(w|\theta_d)} \tag{18}$$

where the summation is taken over the top 50 terms in the relevance model. We used Dirichlet smoothing of document models with $\lambda = 1000$ for all experiments. These parameters demonstrate expected behavior and performance on our collection. Results using query-likelihood and relevance models will be indicated as QL and RM, respectively.

5.2.2 Okapi Scores

Although the majority of our experiments focus on regularizing language modeling scores, this framework can be applied to regularize scores from arbitrary retrieval methods, given some affinity matrix. We therefore conducted experiments studying the regularization of Okapi BM25 scores

[19]. We use the implementation in the Lemur toolkit with default parameter settings.

We use the simple cosine similarity between documents to define our affinity matrix. We could further improve performance by applying a dampening function on top of the cosine measure; adapting the cosine measure in this way has been demonstrated to improve results by acting as a soft nearest-neighbor threshold [28, pages 9-19].

This constitutes a total of six regularization experiments: trec12/QL, trec12/RM, trec12/okapi, robust/QL, robust/RM, robust/okapi. For each of these experiments, we will evaluate various regularization pool sizes. We normalized all scores using a shift-and-scale process.

5.3 Results and Discussion

5.3.1 Regularizing Language Model Scores

Table 2 presents the results for trec12 and Table 3 presents the results for robust.

We first note that regularizing all 1000 documents in the initial retrieval in all cases but one results in a significant improvement in mean average precision. At 1000 documents, these improvements occur at most recall points for QL. The one data set (robust/RM) for which regularizing 1000 documents does not improve mean average precision only sees improvements at larger recall points. This result may be explained by the difficulty of these queries. Notice that baseline RM for the robust data set under-performs regularized QL scores for low recall levels. We believe that if we regularized QL scores over a larger pool, this improvement would become more pronounced.

In general regularizing RM scores only affects higher recall points, indicating that predicted non-relevant documents are being boosted up by related predicted relevant documents. At the same time, these predicted relevant documents are only slightly affected by the process.

We found that α was stable between partitions in our cross-validation but sensitive to retrieval algorithm. For example, the runs using trec12/QL usually had $\alpha = 0.60$ while trec12/RM tended toward $\alpha = 0.30$. The other collection demonstrated similar behavior. A higher α value indicates a more aggressive regularization. The reason for lower values with RM may be in redundancy with pseudo-relevance feedback.

As expected, varying pool size affects only the performance of the subset being regularized. This result implies that there is a tradeoff between the amount of improvement in mean average precision and the computation required for the regularization. We stopped at 1000 documents for practical reasons but believe that performance will improve further when considering more documents.

5.3.2 Regularizing Okapi Scores

In Table 4, we demonstrate the regularization of Okapi retrieval scores. The trends observed for regularization of language models generalize to Okapi scores. This is surprising given that the system only tuned the regularization parameter, α, in training.

6. RELATED WORK

In the preparation of the final draft of this paper, we became aware of unpublished work presenting many ideas related to our algorithm [15]. Although the algorithms are similar, our experiments cover a wider range of topics and collections. We have also placed regularization in the context of well-known information retrieval algorithms.

The majority of techniques in this document are related to work in graph-based methods for semi-supervised learning [3]. Most of these methods are applied to classification tasks where the starting vector y is composed of values in {-1,1,0} for negative, positive, or unlabeled documents, respectively. While there have been methods proposed for incorporating external classifications [28] and strictly positive labels [27], our experiments demonstrate the use of regularization in a true retrieval scenario with highly skewed class distributions and no training instances.

As mentioned earlier, our work is also closely related to cluster-based information retrieval [9, 14]. These techniques construct a lower-dimensional structure of the corpus or initial retrieval. These clusters are then scored according to the query. The original document score and the cluster score are usually interpolated to get a new score. These techniques assume the existence of a very specific lower-dimensional spaces when computing the new score. Our technique relaxes the impact of this assumption by focusing on the local structure; the global behavior is a product of this local analysis.

When viewed as a graph algorithm, our work is also related to the many spreading activation [2, 11, 21, 25, 5] and inference network [22, 17] retrieval methods. In these systems, terms and documents are handled in the same graph framework and usually only direct relationships such as authors or sources allow inter-document links. By focusing on the document manifold, we pay attention to accurately modeling the data rather than on the scoring documents; the score regularization is a product of the document modeling.

A growing body of work focuses on the exploitation of corpus structure for re-ranking documents [9, 10, 18]. These algorithms usually can be interpreted in the generalized regularization framework presented in Section 4. Although related, this work does not explicitly use solutions presented in Section 2. Furthermore, many of these algorithms, inspired by PageRank, operate on directed graphs while our affinity measure is symmetric. The study of graph directedness on such algorithms remains an open research area.

7. CONCLUSION AND FUTURE WORK

We have presented a framework for improving document retrieval scores under a regularization framework. We are considering several extensions. Our experiments exclusively deal with regularizing scores of some relatively small set of retrieved documents. We believe that the improvement in mean average precision with growing regularization pools indicates that doing regularization of even larger—perhaps corpus-level—pools will provide even greater improvements.

8. ACKNOWLEDGMENTS

We would like to thank the various reviewers for helpful feedback. Additionally, many of the experiments in this paper would not have been tractable without Andre Gauthier's technical assistance. This work was supported in part by the Center for Intelligent Information Retrieval and in part by SPAWARSYSCEN-SD grant number N66001-02-1-8903. Any opinions, findings and conclusions or recommendations expressed in this material are of the author and do not necessarily reflect those of the sponsor.

trec12/QL

	0	100	250	500	1000
0.00	0.7532	0.7639	0.7670	0.7493	0.7241
0.10	0.4825	**0.5033**	**0.5059**	**0.5073**	**0.5094**
0.20	0.3997	**0.4124**	**0.4264**	**0.4333**	**0.4349**
0.30	0.3346	**0.3413**	**0.3559**	**0.3635**	**0.3706**
0.40	0.2763	0.2786	**0.2882**	**0.2998**	**0.3123**
0.50	0.2314	0.2312	**0.2433**	**0.2491**	**0.2660**
0.60	0.1816	0.1818	**0.1891**	**0.1927**	**0.2037**
0.70	0.1240	0.1246	**0.1277**	**0.1346**	**0.1414**
0.80	0.0779	0.0782	0.0799	**0.0864**	**0.0931**
0.90	0.0345	0.0345	0.0364	**0.0390**	**0.0439**
1.00	0.0054	0.0054	0.0054	0.0051	0.0047
map	0.2413	**0.2467**	**0.2526**	**0.2576**	**0.2635**

trec12/RM

	0	100	250	500	1000
0.00	0.7561	0.7550	0.7548	0.7555	0.7492
0.10	0.5347	0.5342	0.5337	0.5314	0.5313
0.20	0.4559	0.4552	0.4551	0.4575	0.4585
0.30	0.3889	0.3884	0.3900	0.3921	0.3931
0.40	0.3350	**0.3357**	**0.3371**	**0.3378**	**0.3414**
0.50	0.2793	0.2791	0.2805	**0.2824**	**0.2862**
0.60	0.2286	0.2286	0.2293	**0.2315**	**0.2362**
0.70	0.1664	0.1665	0.1664	**0.1679**	**0.1726**
0.80	0.1048	0.1051	0.1053	0.1061	**0.1085**
0.90	0.0517	0.0517	0.0520	0.0528	**0.0543**
1.00	0.0019	0.0019	0.0021	0.0023	0.0021
map	0.2836	0.2835	0.2836	0.2845	**0.2866**

Table 2: Improvement in QL (left) and RM (right) scores as a function of the number of regularized documents (trec12). Bold numbers indicate statistically significant improvements in performance using the Wilcoxon test ($p < 0.05$).

robust/QL

	0	100	250	500	1000
0.00	0.7401	0.7291	0.7295	0.7321	0.7188
0.10	0.5174	**0.5261**	**0.5242**	**0.5263**	**0.5239**
0.20	0.4097	**0.4190**	**0.4230**	**0.4242**	**0.4226**
0.30	0.3258	**0.3331**	**0.3347**	**0.3344**	**0.3402**
0.40	0.2563	**0.2652**	**0.2688**	**0.2709**	**0.2790**
0.50	0.2168	**0.2232**	**0.2264**	**0.2273**	**0.2320**
0.60	0.1632	**0.1667**	**0.1703**	**0.1722**	**0.1773**
0.70	0.1311	**0.1359**	**0.1405**	**0.1419**	**0.1454**
0.80	0.0912	0.0919	**0.0922**	**0.0945**	**0.0979**
0.90	0.0620	0.0636	0.0650	**0.0677**	**0.0707**
1.00	0.0323	0.0334	0.0335	0.0342	0.0349
map	0.2444	**0.2492**	**0.2513**	**0.2531**	**0.2548**

robust/RM

	0	100	250	500	1000
0.00	0.6877	*0.6844*	*0.6845*	*0.6882*	0.6903
0.10	0.5189	0.5182	*0.5167*	0.5156	*0.5144*
0.20	0.4223	0.4213	*0.4207*	0.4221	0.4199
0.30	0.3512	0.3500	*0.3493*	0.3473	0.3481
0.40	0.2974	*0.2959*	*0.2942*	0.2965	0.2974
0.50	0.2547	0.2541	0.2544	**0.2554**	**0.2569**
0.60	0.2032	0.2034	0.2040	**0.2048**	**0.2068**
0.70	0.1612	0.1612	0.1612	**0.1622**	**0.1635**
0.80	0.1112	0.1118	**0.1122**	**0.1124**	**0.1136**
0.90	0.0644	0.0641	0.0646	**0.0651**	**0.0657**
1.00	0.0237	0.0238	0.0238	0.0238	0.0239
map	0.2626	*0.2619*	*0.2620*	0.2623	0.2625

Table 3: Improvement in QL (left) and RM (right) scores as a function of the number of regularized documents (robust). Bold (italic) numbers indicate statistically significant improvements (degradations) in performance using the Wilcoxon test ($p < 0.05$).

trec12/okapi

	0	100	250	500	1000
0.00	0.7137	0.7323	0.7342	0.7073	0.6936
0.10	0.4733	**0.5062**	**0.5095**	**0.5092**	**0.5142**
0.20	0.3935	**0.4144**	**0.4296**	**0.4399**	**0.4484**
0.30	0.3327	**0.3434**	**0.3577**	**0.3720**	**0.3851**
0.40	0.2722	0.2764	**0.2891**	**0.3054**	**0.3230**
0.50	0.2231	0.2254	**0.2319**	**0.2446**	**0.2644**
0.60	0.1612	0.1619	**0.1669**	**0.1782**	**0.1904**
0.70	0.1089	0.1102	**0.1159**	**0.1237**	**0.1329**
0.80	0.0662	0.0662	0.0673	0.0749	**0.0862**
0.90	0.0179	0.0179	0.0180	0.0190	**0.0215**
1.00	0.0011	0.0011	0.0011	0.0011	0.0014
map	0.2304	**0.2389**	**0.2452**	**0.2527**	**0.2615**

robust/okapi

	0	100	250	500	1000
0.00	0.7167	0.7351	0.7249	0.7173	0.7042
0.10	0.5056	**0.5245**	**0.5271**	**0.5242**	**0.5250**
0.20	0.4027	**0.4290**	**0.4302**	**0.4295**	**0.4330**
0.30	0.3234	**0.3453**	**0.3512**	**0.3544**	**0.3574**
0.40	0.2483	**0.2646**	**0.2797**	**0.2834**	**0.2867**
0.50	0.2084	**0.2161**	**0.2274**	**0.2300**	**0.2345**
0.60	0.1641	**0.1690**	**0.1718**	**0.1743**	**0.1824**
0.70	0.1191	**0.1235**	**0.1285**	**0.1307**	**0.1360**
0.80	0.0741	0.0747	**0.0779**	**0.0811**	**0.0861**
0.90	0.0425	0.0441	**0.0459**	**0.0478**	**0.0512**
1.00	0.0221	0.0234	0.0236	0.0242	**0.0249**
map	0.2333	**0.2462**	**0.2496**	**0.2506**	**0.2532**

Table 4: Improvement in Okapi scores with cosine affinity for trec12 (left) and robust (right) collections as a function of the number of regularized documents. Bold numbers indicate statistically significant improvements in performance using the Wilcoxon test ($p < 0.05$).

9. REFERENCES

[1] J. Allan, J. Callan, K. Collins-Thompson, B. Croft, F. Feng, D. Fisher, J. Lafferty, L. Larkey, T. N. Truong, P. Ogilvie, L. Si, T. Strohman, H. Turtle, L. Yau, and C. Zhai. The lemur toolkit for language modeling and information retrieval. `http://lemurproject.org`.

[2] R. K. Belew. Adaptive information retrieval: using a connectionist representation to retrieve and learn about documents. In *SIGIR '89: Proceedings of the 12th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 11–20, New York, NY, USA, 1989. ACM Press.

[3] M. Belkin and P. Niyogi. Semi-supervised learning on riemannian manifolds. *Mach. Learn.*, 56(1-3):209–239, 2004.

[4] W. B. Croft and J. Lafferty. *Language Modeling for Information Retrieval*. Kluwer Academic Publishing, 2003.

[5] W. B. Croft, T. J. Lucia, and P. R. Cohen. Retrieving documents by plausible inference: a priliminary study. In *SIGIR '88: Proceedings of the 11th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 481–494, New York, NY, USA, 1988. ACM Press.

[6] D. K. Harman. The first text retrieval conference (trec-1) rockville, md, u.s.a., 4-6 november, 1992. *Inf. Process. Manage.*, 29(4):411–414, 1993.

[7] N. Jardine and C. J. V. Rijsbergen. The use of hierarchic clustering in information retrieval. *Information Storage and Retrieval*, 7:217–240, 1971.

[8] R. Krovetz. Viewing morphology as an inference process. In *SIGIR '93: Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 191–202, New York, NY, USA, 1993. ACM Press.

[9] O. Kurland and L. Lee. Corpus structure, language models, and ad hoc information retrieval. In *SIGIR '04: Proceedings of the 27th annual international conference on Research and development in information retrieval*, pages 194–201, New York, NY, USA, 2004. ACM Press.

[10] O. Kurland and L. Lee. Pagerank without hyperlinks: Structural re-ranking using links induced by language models. In *SIGIR '05: Proceedings of the 28th annual international conference on Research and development in information retrieval*, 2005.

[11] K. L. Kwok. A neural network for probabilistic information retrieval. In *SIGIR '89: Proceedings of the 12th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 21–30, New York, NY, USA, 1989. ACM Press.

[12] J. Lafferty and G. Lebanon. Diffusion kernels on statistical manifolds. *J. Mach. Learn. Res.*, 6:129–163, 2005.

[13] V. Lavrenko. *A Generative Theory of Relevance*. PhD thesis, University of Massachusetts, 2004.

[14] X. Liu and W. B. Croft. Cluster-based retrieval using language models. In *SIGIR '04: Proceedings of the 27th annual international conference on Research and development in information retrieval*, pages 186–193, New York, NY, USA, 2004. ACM Press.

[15] I. Matveeva. Text representation with the locality preserving projection algorithm for information retrieval task. Master's thesis, University of Chicago, 2004.

[16] A. K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/ mccallum/bow, 1996.

[17] D. Metzler and W. B. Croft. Combining the language model and inference network approaches to retrieval. *Inf. Process. Manage.*, 40(5):735–750, 2004.

[18] T. Qin, T.-Y. Liu, X.-D. Zhang, Z. Chen, and W.-Y. Ma. A study of relevance propagation for web search. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 408–415, New York, NY, USA, 2005. ACM Press.

[19] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 232–241, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[20] J. J. Rocchio. *The SMART Retrieval System: Experiments in Automatic Document Processing*, chapter Relevance Feedback in Information Retrieval, pages 313–323. Prentice-Hall Inc., 1971.

[21] G. Salton and C. Buckley. On the use of spreading activation methods in automatic information. In *SIGIR '88: Proceedings of the 11th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 147–160, New York, NY, USA, 1988. ACM Press.

[22] H. Turtle and W. B. Croft. Inference networks for document retrieval. In *SIGIR '90: Proceedings of the 13th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 1–24, New York, NY, USA, 1990. ACM Press.

[23] U. von Luxburg, O. Bousquet, and M. Belkin. On the convergence of spectral clustering on random samples: The normalized case. In *Proceedings of the 17th Annual Conference on Learning Theory*, pages 457–471, Berlin, 2004. Springer.

[24] E. Voorhees. Overview of the trec 2004 robust track. In *Proceedings of the 13th Text REtrieval Conference (TREC 2004)*, 2004.

[25] R. Wilkinson and P. Hingston. Using the cosine measure in a neural network for document retrieval. In *SIGIR '91: Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 202–210, New York, NY, USA, 1991. ACM Press.

[26] J. Xu and W. B. Croft. Cluster-based language models for distributed retrieval. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 254–261, New York, NY, USA, 1999. ACM Press.

[27] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In L. S. Thrun, S. and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, volume 16, pages 169–176, Cambridge, MA, USA, 2004. MIT Press.

[28] X. Zhu. *Semi-Supervised Learning with Graphs*. PhD thesis, Carnegie Mellon University, 2005. CMU-LTI-05-192.