Inf Retrieval (2007) 10:531–562
DOI 10.1007/s10791-007-9034-8

# Regularizing query-based retrieval scores

**Fernando Diaz**

Received: 20 February 2007 / Accepted: 20 August 2007 / Published online: 21 September 2007


**Abstract** We adapt the cluster hypothesis for score-based information retrieval by claiming that closely related documents should have similar scores. Given a retrieval from an arbitrary system, we describe an algorithm which directly optimizes this objective by adjusting retrieval scores so that topically related documents receive similar scores. We refer to this process as score regularization. Because score regularization operates on retrieval scores, regardless of their origin, we can apply the technique to arbitrary initial retrieval rankings. Document rankings derived from regularized scores, when compared to rankings derived from un-regularized scores, consistently and significantly result in improved performance given a variety of baseline retrieval algorithms. We also present several proofs demonstrating that regularization generalizes methods such as pseudo-relevance feedback, document expansion, and cluster-based retrieval. Because of these strong empirical and theoretical results, we argue for the adoption of score regularization as general design principle or post-processing step for information retrieval systems.

**Keywords** Regularization · Cluster hypothesis · Cluster-based retrieval · Pseudo-relevance feedback · Query expansion · Document expansion

## 1 Introduction

In information retrieval, a user presents a query to a computer; the computer then returns documents in a corpus relevant to the user's query. A user familiar with the topic may be able to supply example relevant and non-relevant documents. More often, a user is unfamiliar with the topic and possesses no example documents. In this situation, the user provides a short, natural language query to the computer. We refer to this situation as *query-based information retrieval*.

F. Diaz (✉)
Department of Computer Science, University of Massachusetts-Amherst,
140 Governor's Drive, Amherst, MA 01003-4610, USA
e-mail: fdiaz@cs.umass.edu

A *set retrieval model* assigns a binary prediction of relevance to each document in the collection. The user then scans those documents predicted to be relevant. We can see this as a mapping or *function* from documents in the collection to a binary value. Mathematically, given a query, $q$, a set retrieval model provides a function, $f_q : \mathcal{D} \rightarrow \{0, 1\}$, from documents to labels; we refer to $f_q$ as the *initial score function* for a particular query. The argument of this function is the retrieval system's representation of a document. The values of the function provide the system's labeling of the documents. Notice that we index functions by the query. We note this to emphasize the fact that, in information retrieval, the score function over all documents will be different for each query. Although we drop the index for notational convenience, the reader should keep in mind that this is a function for a particular query.

A *ranked retrieval model* assigns some rank or score to each document in the collection and ranks documents according to the score. The user then scans the documents according to the ranking. The score function for a ranked retrieval model maps documents to real values. Given a query, $q$, the model provides a function, $f_q : \mathcal{D} \rightarrow \Re$, from documents to scores. The values of the function provide the desired ranking of the documents. There exist many ranked retrieval models based on geometry (e.g., the vector space model (Salton et al. 1975)) and probability (e.g., the probabilistic model (Robertson et al. 1981), inference networks (Turtle and Croft 1990), and language modeling (Croft and Lafferty 2003)). This paper examines the behavior of score functions for ranked retrieval models with respect to the geometry of the underlying domain, $\mathcal{D}$.

One way to describe a function, regardless of its domain, is by its *smoothness*. The smoothness of a function might be measured, for example, by its continuity, as in Lipschitz continuity. In many situations, we prefer functions which exhibit smoothness. For example, consider the one-dimensional functions in Fig. 1. If we assume that local consistency or continuity in the function is desirable, then the function depicted in the Fig. 1b is preferable because it is smoother.

If only presented with the function in Fig. 1a, then we can procedurally modify the function to better satisfy our preference for smooth functions. The result may be the function in Fig. 1b. Post-processing a function is one way to perform *regularization* (Chen and Haykin 2002). In our work, we regularize initial score functions. Because our analysis and regularization is local to the highest scored documents, we refer to this process as *local score regularization*.

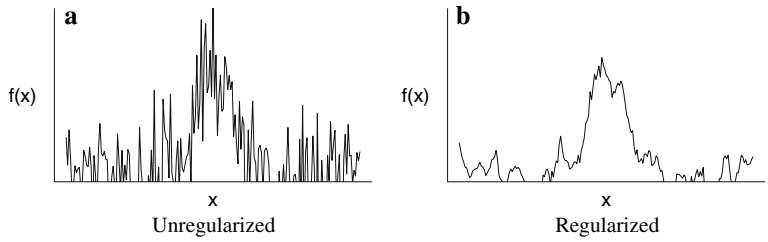


**Fig. 1** Functions in one dimension. Each value on the horizontal axis may, for example, represent a one-dimensional classification code such as a linear library ordering of books. The functions in these figures assign a value to each point on the real line and may represent relevance. If a set of functions are intended to describe the same phenomenon or signal, we can develop criteria for preferring one function over another. If we prefer smoother function, we would dismiss the function in a in favor of the function in (**b**). The process of smoothing the function in (**a**) into the function in (**b**) is a type of regularization

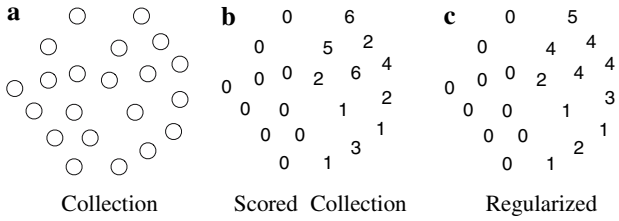


**Fig. 2** Regularizing retrieval scores. Documents in a collection can often be embedded in a vector space as shown in (**a**). When presented with a query, a retrieval system provides scores for all of the documents in the collection (**b**). Score regularization refers to the process of smoothing out the retrieval function such that neighboring documents receive similar scores (**c**)

When our domain was the real line, we wanted the value of the function at two points, $f(x_1)$ and $f(x_2)$, to be similar if the distance between the two points, $|x_1 - x_2|$, was small. In information retrieval, our domain is the set of documents and we want the value of the function for two documents to be similar if the "distance between two documents" is small. We adopt a topic-based distance and consider two documents close if they share the same or similar topics. We will refer to topical closeness as topical *affinity*. Affinity between documents can be measured using, for example, inter-document cosine similarity. We would like two documents which share the same topic to receive similar scores. We depict this graphically in Fig. 2a for documents in a two-dimensional embedding space. When presented with a query, the retrieval system computes scores for each document in this space (Fig. 2b); this is our initial score function. We regularize a function into order to improve the consistency of scores between neighboring documents. This is depicted graphically in Fig. 2c where the value of the function is smoother in the document space. Of course, realistic collections often cannot be visualized like this two-dimensional example. Nevertheless, the fundamental regularization process remains roughly the same.

There is an interesting connection here to the cluster hypothesis. The cluster hypothesis states: *closely related documents tend to be relevant to the same request* (Jardine and van Rijsbergen 1971). In regularization, we extend this hypothesis to score-based retrieval: *given a query, closely related documents should have similar scores*.[1] In this paper, we present theoretical and empirical arguments for why score regularity should be adopted as a design principal for information retrieval systems. Because we formally define this objective and optimize it directly, we view score regularization as being in the spirit of axiomatic retrieval (Fang and Zhai 2005).

Why might systems produce scores which fail to conform to the cluster hypothesis? Query-based information retrieval systems often score documents independently. The score of document $a$ may be computed by examining query term matches, document length, and global collection statistics. Many initial retrieval functions operate this way (Fang et al. 2004). Once computed, a system rarely compares the score of $a$ to the score of a topically-related document $b$. With some exceptions, the correlation of document scores is largely been ignored, leaving room for improvement through regularization.

Broadly, this paper contributes the following results,

1. An algorithm, local score regularization, designed to adjust retrieval scores to respect inter-document consistency (Sect. 4)

[1] Baliński and Daniłowicz (2005) recently proposed a similar score-based objective. Though a solution is presented, we are not aware of any experimental results or connections to previous models we describe here.

2. A reduction of several well-known retrieval methods to score regularization (Sect. 5)
3. Experiments demonstrating strong and consistent performance improvements when score regularization is applied to arbitrary retrieval methods (Sect. 7)

This paper can be broken into three parts: a description of score regularization, a discussion of relationships to other techniques, and experimental results. In the first part, we will describe the score regularization algorithm. Because we use a variety of mathematical conventions, we review these conventions and formally state our problem in Sect. 2. Document affinity and relatedness are critical to regularization. We present a graph-based approach to regularization in Sect. 3. We then describe the general regularization framework in Sect. 4. In the second part of our paper, we place regularization in the context of previous work. Because regularization has an interesting relationship to several classic information retrieval techniques, we devote Sect. 5 to reductions of several well-known techniques to score regularization. In the third part of our paper, we present experimental arguments for score regularization. We describe our experimental setup in Sect. 6. The results of these experiments are presented in Sect. 7 and discussed in Sect. 8. We conclude in Sect. 9.

## 2 Preliminaries

### 2.1 Notation

We adopt vector and matrix notational convention from previous work (Petersen and Pedersen 2005). These conventions are reviewed in Table 1.

### 2.2 Definitions

A *collection* is a set of $n$ documents which exist in an $m$-dimensional vector space where $m$ is the size of the vocabulary and elements of the vectors represent the frequency of the term in the document. We define for each document $1 \leq i \leq n$ a column vector, $\mathbf{d}_i$, where each element of the vector represents the frequency of the term in document $i$; we refer to this as the *document vector*. These document vectors may be normalized by their $L_1$ or $L_2$ norm. We will attempt to make norms, if any, clear in context. Transposing and stacking up the $n$ document vectors defines the $n \times m$ collection matrix $\mathbf{C}$.

We define other symbols in Table 1. Elaborations of definitions will occur when notation is introduced.

### 2.3 Problem statement

We now formally define the regularization task. The **input** is a vector of document scores. Although the system usually scores all $n$ documents in the collection, we consider only the top $\tilde{n}$ scores. The $\tilde{n} \times 1$ vector, $\mathbf{y}$, represents these scores. This vector may be normalized if desired. For example, we normalize this vector to have zero-mean and unit variance. The **output** is the vector of regularized scores represented by the $\tilde{n} \times 1$ vector $\mathbf{f}$. The objective is to define a regularization process which results in a superior ranking of the documents represented in $\mathbf{y}$, given some evaluation measure. In our work, we use mean average

**Table 1** Definition of symbols

| | |
|---|---|
| $\mathbf{A}$ | matrix |
| $\mathbf{A}_i$ | the $i$th matrix |
| $A_{ij}$ | element $(i, j)$ of matrix $\mathbf{A}$ |
| $\mathbf{a}$ | vector |
| $\mathbf{a}_i$ | the $i$th vector |
| $a_i$ | element $i$ of vector $\mathbf{a}$ |
| $a$ | scalar |
| $f(\mathbf{A})$ | element-wise function of $\mathbf{A}$ |
| $\mathbf{A}^{1/2}$ | element-wise square root |
| $\mathbf{A}^{-1}$ | matrix inverse |
| $\mathbf{A}^{\mathrm{T}}$ | matrix transpose |
| $\|\mathbf{a}\|_p$ | $\left(\sum_{i=1}^{n} \|a_i\|^p\right)^{1/p}$; $L_p$ norm of the vector $\mathbf{a}$ |
| $n$ | number of documents |
| $\tilde{n}$ | number of documents to regularize |
| $m$ | number of terms |
| $\mathbf{C}$ | $n \times m$ collection matrix; elements are the model-specific term weights |
| $\mathbf{d}_i$ | row $i$ of $\mathbf{C}$ as a $m \times 1$ column vector |
| $\mathbf{w}_i$ | column $i$ of $\mathbf{C}$ |
| $\mathbf{l}$ | $n \times 1$ vector of document lengths |
| $\mathbf{c}$ | $m \times 1$ vector of term document frequencies |
| $\mathbf{A}$ | $n \times n$ document affinity matrix |
| $\mathbf{W}$ | nearest neighbor graph based on $\mathbf{A}$ |
| $\mathbf{y}$ | $n \times 1$ initial score vector |
| $\mathbf{f}$ | $n \times 1$ regularized score vector |
| $\mathbf{U}$ | $m \times k$ matrix of cluster vectors |
| $\mathbf{V}$ | $k \times n$ matrix of documents embedded into $k$ dimensions |
| $\mathbf{y}_c$ | $k \times 1$ cluster score vector |
| $\mathbf{W}_e$ | $n \times n$ graph based on expanded documents |
| $\mathbf{y}_e$ | $n \times 1$ vector of scores for expanded documents |
| $\mathbf{\Delta}$ | $n \times n$ Laplacian on $\mathbf{W}$ |
| $\mathbf{E}_k$ | $n \times k$ matrix of top $k$ eigenvectors of $\mathbf{W}$ |
| $\mathbf{e}$ | column vector of all 1's |
| $\mathbf{I}$ | identity matrix |

precision (MAP) as the evaluation metric. MAP provides a standard and stable evaluation metric (Buckley and Voorhees 2000).

## 3 Computing inter-document affinity

In Fig. 2, we depicted documents existing in some space where proximity related to topical affinity. Our representations will never be as simple as those in our toy example. We now turn to describing one method for describing the relationship between documents. Our approach will be to construct a content-based graph of the corpus. In this graph, nodes represent documents and edges represent the similarity between document vectors. We will

build this graph in two steps: (1) compute the similarity between all pairs of documents using a standard text-based method and (2) add edges to the graph using the $k$-nearest neighbors of each document. In Sects. 3.1 and 3.2, we describe two measures of similarity between document vectors. The similarity between all pairs of $\tilde{n}$ documents can be represented by $\tilde{n} \times \tilde{n}$ matrix $\mathbf{A}$. In Sect. 3.3, we construct a nearest neighbor graph using the similarity information.

### 3.1 Cosine similarity

If we assume that each document vector, $\mathbf{d}_i$, is normalized by its $L_2$ norm, then each document can be placed on an $m$-dimensional hypersphere (Salton 1968). The inner product between document vectors determines affinity,

$$\begin{aligned} A_{ij} &= \langle \mathbf{d}_i, \mathbf{d}_j \rangle \\ &= \mathbf{d}_i^{\mathrm{T}} \mathbf{d}_j \end{aligned} \tag{1}$$

which is equivalent to the standard cosine similarity measure. The $\tilde{n} \times \tilde{n}$ affinity matrix is defined by,

$$\mathbf{A} = \mathbf{C}\mathbf{C}^{\mathrm{T}} \tag{2}$$

where each element of the matrix defines the symmetric affinity between two documents.

### 3.2 Language model similarity

The language modeling perspective of information retrieval treats the text occurring in a document as having been generated by an unknown probabilistic model (Croft and Lafferty 2003). If we constrain this model to have a certain form, then we can then apply statistical methods for estimating the parameters of the model given the text occurring in a document. Although many different models have been proposed, practitioners often assume that each document is generated by a unique multinomial over terms. The parameters of these $n$ multinomials can be estimated in a number of ways but in this section we will focus on the maximum likelihood estimate. If we let $P(w|\theta_d)$ be a multinomial distribution over $m$ terms, then the maximum likelihood estimate is defined as

$$P(w_i|\theta_d) = \frac{d_i}{\|\mathbf{d}\|_1} \tag{3}$$

Therefore, in this section, we consider $\mathbf{d}$ to be an $L_1$-normalized vector of term frequencies which is equivalent to the maximum likelihood estimate.

For language models, we can adopt a measure of similarity between multinomials. 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 (Lafferty and Lebanon 2005). We adopt this measure because it is symmetric (allowing closed form solutions in Sect. 4.3) and has been used successfully for text clustering and classification tasks. This affinity measure between two distributions, $\mathbf{d}_i$ and $\mathbf{d}_j$, is motivated by Fisher information metric and defined as,

$$A_{ij} = \mathcal{K}(\mathbf{d}_i, \mathbf{d}_j) = \exp\left(-t^{-1} \arccos^2\left\langle \mathbf{d}_i^{1/2}, \mathbf{d}_j^{1/2} \right\rangle\right) \tag{4}$$

where $t$ is a parameter controlling the decay of the affinity. The selection of a value for $t$ will be clarified in Sect. 6.2.5. In fact, when two multinomials are very similar, the value of the diffusion kernel approximates that of the Kullback–Leibler divergence.

The $\tilde{n} \times \tilde{n}$ affinity matrix is defined by,

$$\mathbf{A} = \exp(-t^{-1} \arccos^2(\mathbf{C}\mathbf{C}^{\mathrm{T}})) \tag{5}$$

Notice that, besides the normalization of the vectors, this is equivalent to applying a soft threshold to Eq. 2.

### 3.3 Graph construction

For the top $\tilde{n}$ documents, we compute the complete $\tilde{n} \times \tilde{n}$ affinity matrix, $\mathbf{A}$; however, 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. Consider the clusters of points in Fig. 3.

We would like the distances to respect the clustering; documents in the same cluster should have smaller distances to each other than documents in different clusters. The ambient distance (Fig. 3a) clearly does not satisfy this property. Using the geodesic distance (Fig. 3b) seems more appropriate. A nearest neighbor graph preserves the type of geodesic distances we desire in here. For example, an $\tilde{n} \times \tilde{n}$ matrix, $\mathbf{W}$, may only include the affinities to the $k$-nearest neighbors for each document from the affinity matrix, $\mathbf{A}$, and zero otherwise. Constructing a document *affinity graph* captures the lower-dimensional document manifold and has demonstrated usefulness for text classification tasks (Belkin and Niyogi 2004). We explore the appropriateness of this assumption in our experiments.

A retrieval system theoretically provides scores for all $n$ documents in the collection. To perform global analysis, our method would need to construct a graph including all $n$ documents. Computational constraints prevent building the complete affinity matrix. We therefore build graphs considering the top $\tilde{n}$ documents from the initial retrieval. This

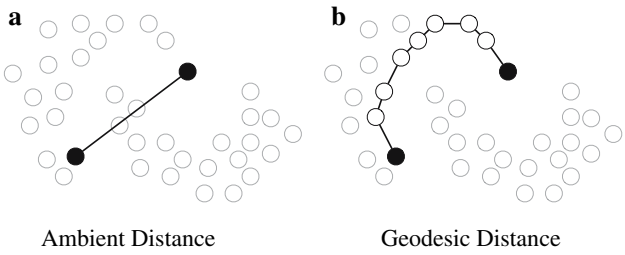


**Fig. 3** Ambient versus geodesic distance. The data naturally form two clusters. We would like the distances between points in the same cluster to be smaller than distances between points in different clusters. Points from a foreign cluster appear closer than points in the same cluster when using the ambient distance (**a**). This problem is mitigated when using the geodesic distance (**b**)

*query-biased* graph-construction procedure is depicted in Fig. 4. We justify this methodology by noting that the score function will be flat for the majority of the collection since the majority of the collection is non-relevant. Query-biased graphs focus regularization on the portion of the document graph most likely to contain relevant documents.

By using a graph, we assume the presence of a lower-dimensional manifold underlying the documents in the space; however, we should, at this point, stress a few problems with this assumption. First, there is no explicit evidence that the documents from the initial retrieval lie on a lower-dimensional manifold. We cannot visualize the documents in their ambient space and observe some lower-dimensional structure. Implicitly, though, the success of cluster-based retrieval methods suggests that there probably exists some topical substructure (Liu and Croft 2004; Xu and Croft 1999). From a theoretical perspective, methods such as manifold regularization normally assume a uniform sampling on the manifold (Belkin and Niyogi 2005). We need this assumption in order to, for example, demonstrate the convergence of the graph Laplacian in Sect. 4.1 to the continuous Laplacian. However, we cannot assume that topics are equally represented. Some topics will, in general, be represented by fewer documents than other topics. If we use a (biased) sample from the initial retrieval, this non-uniformity will be exacerbated. Therefore, whenever possible, we have attempted to use methods robust to violations of the sampling assumption (see Sect. 4.1).

## 4 Local score regularization

In this section, we will present a regularization method which applies previous results from machine learning (Zhou et al. 2004). We will review these results in the vocabulary of information retrieval. More thorough derivations can be found in cited publications.

Given the initial scores as a vector, $\mathbf{y}$, we would like to compute a set of regularized scores, $\mathbf{f}$, for these same documents. To accomplish this, we propose two contending objectives: score consistency between related documents and score consistency with the initial retrieval. These two objectives are depicted graphically for a one-dimensional function in Fig. 5. Let $\mathcal{S}(\mathbf{f})$ be a cost function associated with the inter-document consistency of the scores, $\mathbf{f}$; if related documents have very inconsistent scores, then the value of this function will be high. Let $\mathcal{E}(\mathbf{f}, \mathbf{y})$ 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. For mathematical simplicity, we use a linear combination of these objectives for our composite objective function,

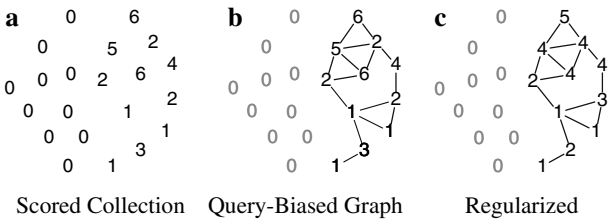


**Fig. 4** Graph-based score regularization. The input of any regularization algorithm is a set of scores over the documents in the collection (**a**). Our algorithm first builds a nearest-neighbor graph using only the top $n$ documents (**b**). We then apply regularization based on the Laplacian of this graph (**c**)

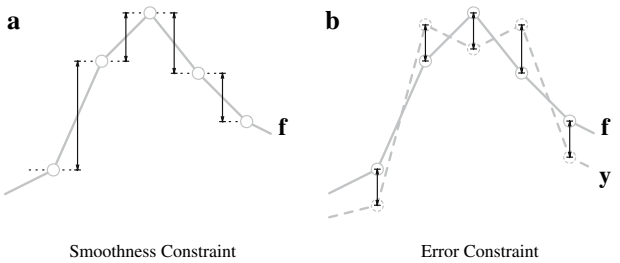


**Fig. 5** Smoothness and error constraints for a function on a linear graph. In (**a**), the smoothness constraint penalizes functions where neighboring nodes in **f** receive different values. In (**b**), the error constraint penalizes functions where nodes in **f** receive values different from the corresponding values in **y**

$$\mathcal{Q}(\mathbf{f}, \mathbf{y}) = \mathcal{S}(\mathbf{f}) + \mu\mathcal{E}(\mathbf{f}, \mathbf{y}) \tag{6}$$

where $\mu$ is a regularization parameter allowing us to control how much weight to place on inter-document smoothing versus consistency with the original score.[2]

### 4.1 Measuring inter-document consistency

Inter-document relatedness is represented by the graph, $\mathbf{W}$, defined in Sect. 3.3 where $W_{ij}$ represents the affinity between documents $i$ and $j$. We define our graph so that there are no self-loops ($W_{ii} = 0$). A set of scores is considered smooth if related documents have similar scores. In order to quantify smoothness, we define the cost function, $\mathcal{S}(\mathbf{f})$, which penalizes inconsistency between related documents,

$$\mathcal{S}(\mathbf{f}) = \sum_{i,j=1}^{\tilde{n}} W_{ij}\left(f_i - f_j\right)^2 \tag{7}$$

We measure inconsistency using the weighted difference between scores of neighboring documents.[3]

In spectral graph theory, Eq. 7 is known as the Dirichlet sum (Chung 1997). We can rewrite the Dirichlet sum in matrix notation,

$$\sum_{i,j=1}^{\tilde{n}} W_{ij}\left(f_i - f_j\right)^2 = \mathbf{f}^{\mathrm{T}}(\mathbf{D} - \mathbf{W})\mathbf{f} \tag{8}$$

where $\mathbf{D}$ is the diagonal matrix defined as $D_{ii} = \sum_{j=1}^{\tilde{n}} W_{ij}$. The matrix $(\mathbf{D} - \mathbf{W})$ is known as the *combinatorial Laplacian* which we represent by $\mathbf{\Delta}_C$. The graph Laplacian can be viewed as the discrete analog of the Laplace–Beltrami operator. Because the Laplacian can be used to compute the smoothness of a function, we may abstract $\mathbf{\Delta}_C$ and replace it with

[2] These functions operate on the entire vector **f** as opposed to element-wise.

[3] The local, discrete Lipschitz constant for a document, $i$, can be thought of as $\max_j (W_{ij}\|f_i - f_j\|)$. Although similar, the local Lipschitz measure is much less forgiving to discontinuities in a function. Because our retrieval function can be thought of as a very peaked or spiky function due to the paucity of relevant documents, we adopt the Laplacian-based measure.

alternative formulations of the Laplacian which offer alternative measures of smoothness. For example, the *normalized Laplacian* is defined as,

$$\begin{aligned}\mathbf{\Delta}_N &= \mathbf{D}^{-1/2}\mathbf{\Delta}_C\mathbf{D}^{-1/2} \\ &= \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}\end{aligned} \tag{9}$$

measures the degree-normalized smoothness as,

$$\mathbf{f}^{\mathrm{T}}\mathbf{\Delta}_N\mathbf{f}^{\mathrm{T}} = \sum_{i,j=1}^{\tilde{n}} \frac{W_{ij}}{D_{ii}D_{jj}}\left(f_i - f_j\right)^2 \tag{10}$$

The *approximate Laplace–Beltrami operator* is a variation of the normalized Laplacian which uses a modified affinity matrix (Lafon 2004). The approximate Laplace–Beltrami operator is defined as,

$$\mathbf{\Delta}_A = \mathbf{I} - \hat{\mathbf{D}}^{-1/2}\hat{\mathbf{W}}\hat{\mathbf{D}}^{-1/2} \tag{11}$$

where we use the adjusted affinity matrix $\hat{\mathbf{W}} = \mathbf{D}^{-1}\mathbf{W}\mathbf{D}^{-1}$ with $\hat{D}_{ii} = \sum_{j=1}^{\tilde{n}} \hat{W}_{ij}$. The approximate Laplace–Beltrami operator theoretically addresses violations of the uniform sampling assumption. Because we were concerned with the violation of this assumption at the end of Sect. 3.3, we adopt the approximate Laplace–Beltrami operator (Eq. 11) in our work. We examine the effect of this choice on the regularization performance in Sect. 7.1.

The value of the objective, $\mathcal{S}(\mathbf{f})$ is small for smooth functions and large for non-smooth function. Unconstrained, however, the function minimizing this objective is the constant function

$$\mathrm{argmin}_{\mathbf{f}}\mathcal{S}(\mathbf{f}) = \mathbf{e}$$

In the next section, we will define a second objective which penalizes regularized scores inordinately inconsistent with the initial retrieval.

### 4.2 Measuring consistency with initial scores

We define a second objective, $\mathcal{E}(\mathbf{f}, \mathbf{y})$, which penalizes inconsistencies between the initial retrieval scores, $\mathbf{y}$, and the regularized scores, $\mathbf{f}$,

$$\mathcal{E}(\mathbf{f}, \mathbf{y}) = \sum_{i=1}^{\tilde{n}} (f_i - y_i)^2 \tag{12}$$

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

### 4.3 Minimizing the objective function

In the previous two sections, we defined two constraints, $\mathcal{S}(\mathbf{f})$ and $\mathcal{E}(\mathbf{f}, \mathbf{y})$, which can be combined as a single objective, $\mathbf{f}$. Formally, we would like to find the optimal set of regularized scores, $\mathbf{f}^*$, such that,

$$\mathbf{f}^* = \mathrm{argmin}_{\mathbf{f}\in\Re^{\tilde{n}}}\mathcal{Q}(\mathbf{f}, \mathbf{y}) \tag{13}$$

In this section, we will describe two solutions, one iterative and one closed-form, to compute the regularized scores $\mathbf{f}^*$.

Our iterative solution to this optimization interpolates the score of a document with the scores of its neighbors. Metaphorically, this process, at each iteration, *diffuses* scores on the document graph. This is accomplished mathematically by defining a diffusion operator, $S$, for each Laplacian.

| | $S$ |
|---|---|
| $\mathbf{\Delta}_C$ | $\mathbf{W}$ |
| $\mathbf{\Delta}_N$ | $\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ |
| $\mathbf{\Delta}_A$ | $\hat{\mathbf{D}}^{-1/2}\hat{\mathbf{W}}\hat{\mathbf{D}}^{-1/2}$ |

Given this operator, the score diffusion process can be formulated as,

$$\mathbf{f}^{t+1} = (1-\alpha)\mathbf{y} + \alpha S\mathbf{f}^t \tag{14}$$

where $\alpha = \frac{1}{1+\mu}$ (Zhou et al. 2004). We can initialize the regularized scores such that $\mathbf{f}^0 = \mathbf{y}$. As $t$ approaches $\infty$, the regularized scores, $\mathbf{f}^t$, converge on the optimal scores, $\mathbf{f}^*$. The iterative diffusion in Eq. 14 provides an intuitive flavor for the solution to our optimization.

In our work, we use the closed form solution to Eq. 13. The optimal regularized scores can be formulated as the solution of matrix operations,

$$\mathbf{f}^* = (1-\alpha)(\alpha\mathbf{\Delta} + (1-\alpha)\mathbf{I})^{-1}\mathbf{y} \tag{15}$$

where $\alpha$ is defined above.

Our final score regularization algorithm is presented in Fig. 6. Note that the affinity matrix computed in Step 1 is used for adding elements to $\mathbf{W}$ in Step 2 and does not define $\mathbf{W}$ itself unless $k = \tilde{n}$.

## 5 Corpus-aware retrieval methods which reduce to instances of iterative score regularization

Several classic retrieval methods can be posed as instances of score regularization. We will be focusing on the relationship between these methods and a single iteration of score regularization (Eq. 14). In previous sections, we considered only the top $\tilde{n} \ll n$ documents

1. compute $\tilde{n} \times \tilde{n}$ affinity matrix
2. add the $k$ nearest neighbors for each document to $\mathbf{W}$
3. compute Laplacian, $\mathbf{\Delta}$
4. $\mathbf{f}^* = (1-\alpha)\,(\alpha\mathbf{\Delta} + (1-\alpha)\mathbf{I})^{-1}\,\mathbf{y}$

| | |
|---|---|
| $\tilde{n}$ | number of document scores to regularize |
| $\mathbf{y}$ | top $\tilde{n}$ initial retrieval scores |
| $k$ | number of neighbors to consider |
| $\alpha$ | parameter favoring inter-document consistency |
| $\mathbf{f}^*$ | regularized scores |

**Fig. 6** Local Score Regularization Algorithm. Inputs are $\tilde{n}$, $\mathbf{y}$, $k$ and $\alpha$. The output is the length $\tilde{n}$ vector of regularized scores, $\mathbf{f}^*$

from some initial retrieval. In this section, we may at times consider every document in the collection (i.e., $\tilde{n} = n$).

For each of the methods in this section, we will be asking ourselves the following question: can the final retrieval scores be computed as a function of the initial retrieval scores and a similarity-based adjacency matrix? If the answer to this question is "yes", then we can state that this method is an instance of score regularization. We present a summary of these results in Table 2.

### 5.1 Vector space model retrieval

In Sect. 3.1, we represented each document as a $L_2$ normalized, length-$m$ vector, $\mathbf{d}$. A query can also be represented by a normalized, length-$m$ vector, $\mathbf{q}$. A document's score is the inner product between its vector and the query vector (i.e., $y_i = \langle \mathbf{d}_i, \mathbf{q} \rangle$).

*Pseudo-relevance feedback* or *query expansion* refers to the technique of building a model out of the top $r$ documents retrieved by the original query. The system then performs a second retrieval using combination of this model and the original query. In the vector space model, the classic Rocchio pseudo-relevance feedback algorithm assumes that the top $r$ documents from the initial retrieval are relevant (Rocchio 1971). Let this *pseudo-relevant* set be $R$ and $r = |R|$. In Rocchio, we linearly combine the vectors of documents in $R$ with the original query vector, $\mathbf{q}$. The modified query, $\tilde{\mathbf{q}}$, is defined as,

$$\tilde{\mathbf{q}} = \mathbf{q} + \frac{\alpha}{r} \sum_{j \in R} \mathbf{d}_j \tag{16}$$

where $\alpha$ is the weight placed on the pseudo-relevant documents. We can then use this new representation to score documents by their similarity to $\tilde{\mathbf{q}}$.

**Theorem 1** *Pseudo-relevance feedback in the vector space model is a form of regularization.*

**Table 2** Comparison of corpus modeling and graph-based algorithms

| | Score |
|---|---|
| *Vector space model* | |
| Query expansion | $\mathbf{Ay} + \mathbf{y}$ |
| Document expansion | $\mathbf{Wy} + \mathbf{y}$ |
| Cluster-based retrieval | $\mathbf{V}^{\mathrm{T}}\mathbf{y}_c + \mathbf{y}$ |
| *Language modeling* | |
| Query expansion | $\mathbf{Ay} + \mathbf{y}$ |
| Document expansion | $\log(\mathbf{AC} + \mathbf{C})\mathbf{q}$ |
| Cluster-based retrieval | $\log(\mathbf{V}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}} + \mathbf{C})\mathbf{q}$ |
| Cluster interpolation | $\mathbf{W}_e\mathbf{y}_e + \mathbf{y}$ |
| *Regularization* | |
| Iterative regularization | $\mathbf{Wy} + \mathbf{y}$ |
| Closed form regularization | $(\alpha\mathbf{\Delta} + (1-\alpha)\mathbf{I})^{-1}\mathbf{y}$ |
| *Laplacian eigenmaps* | $\mathbf{W}_c\mathbf{y}_c$ |
| *PageRank* | $\mathbf{E}_1 \circ \mathbf{y}$ |

Model-specific constants and parameters have been omitted for clarity

*Proof* First, we note that the similarity between a document and the new query can be written as the combination of the original document score and the sum of similarities to the pseudo-relevant set,

$$\begin{aligned}\langle \mathbf{d}_i, \tilde{\mathbf{q}} \rangle &= \left\langle \mathbf{d}_i, \mathbf{q} + \frac{\alpha}{r} \sum_{j \in R} \mathbf{d}_j \right\rangle \\ &= \langle \mathbf{d}_i, \mathbf{q} \rangle + \frac{\alpha}{r} \left\langle \mathbf{d}_i, \sum_{j \in R} \mathbf{d}_j \right\rangle \\ &= \langle \mathbf{d}_i, \mathbf{q} \rangle + \frac{\alpha}{r} \sum_{j \in R} \langle \mathbf{d}_i, \mathbf{d}_j \rangle \end{aligned} \tag{17}$$

Notice here that the first factor in the sum is $y_i$ and the second factor in the sum represents the similarity to the pseudo-relevant documents, $\sum_{j \in R} A_{ij}$. We can rewrite Eq. 17 in terms of matrix operators to compute the new scores for all documents in the collection. This computation is a function of the initial scores and the inner product affinity matrix,

$$\mathbf{f} = \mathbf{y} + \frac{\alpha}{\|\sigma(\mathbf{y})\|_1} \mathbf{A} \sigma(\mathbf{y}) \tag{18}$$

where $\sigma(\mathbf{y}) : \Re^n \rightarrow \Re^n$ is defined as,

$$\sigma(\mathbf{y})_i = \begin{cases} 1 & \text{if } i \in R \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

We compare $\sigma(\mathbf{y})$ to $\mathbf{y}$ in Fig. 7. The $\sigma$ function maps high-ranked documents to pseudo-scores of 1. This behavior replicates the judgment of documents as relevant. From our perspective of score functions, we see that $\sigma$ acts as a hard filter on the signal $\mathbf{y}$. This demonstrates that Rocchio is an instance of score regularization. □

Whereas query expansion incorporates into a query terms from $r$ pseudo-relevant documents, *document expansion* incorporates into a document the terms from its $k$ most similar neighbors (Singhal and Pereira 1999). The modified document, $\tilde{\mathbf{d}}$, is defined as,

$$\tilde{\mathbf{d}}_i = \alpha_D \mathbf{d}_i + \frac{1}{k} \sum_{j \in N(i)} \mathbf{d}_j \tag{20}$$

where $\alpha_D$ is the weight placed on the original document vector. $N$ is the set of $k$ documents most similar to document $i$.

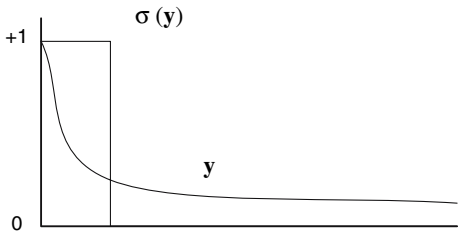


**Fig. 7** Hard weighting function for pseudo-relevance feedback. The horizontal axis represents the documents in decreasing order of $\mathbf{y}$. The function $\sigma(\mathbf{y})$ acts as a filter for pseudo-relevant documents. It sets the score of each of the top $r$ documents to 1

**Theorem 2** *Document expansion in the vector space model is a form of regularization.*

*Proof* Define the binary matrix $\mathbf{W}$ so that each row $i$ contains $k$ non-zero entries for each of the indices in $N(i)$. We can expand all documents in the collection,

$$\tilde{\mathbf{C}} = \alpha_D \mathbf{C} + \frac{1}{k}\mathbf{WC} \tag{21}$$

Given a query vector, we can score the entire collection,

$$\begin{aligned}\mathbf{f} &= \tilde{\mathbf{C}}\mathbf{q} \\ &= (\alpha_D \mathbf{C} + \frac{1}{k}\mathbf{WC})\mathbf{q} \\ &= \alpha_D \mathbf{Cq} + \frac{1}{k}\mathbf{WCq} \\ &= \alpha_D \mathbf{y} + \frac{1}{k}\mathbf{Wy}\end{aligned} \tag{22}$$

The implication here is that the score of an expanded document ($f_i$) is the linear combination of the original score ($y_i$) and the scores of its $k$ neighbors ($\frac{1}{k}\sum_{j \in N(i)} y_i$). This demonstrates that document expansion is a form of regularization. □

We now turn to the dimensionality reduction school of cluster-based retrieval algorithms. In the previous proof, we expanded the entire collection using the matrix $\mathbf{W}$. Clustering techniques such as Latent Semantic Indexing (LSI) can also be used to expand documents (Deerwester et al. 1990). LSI-style techniques use two auxiliary matrices: $\mathbf{V}$ is the $k \times n$ matrix embedding documents in the $k$-dimensional space and $\mathbf{U}$ is $m \times k$ representations of the dimensions in the ambient space. Oftentimes, queries are processed by projecting them into the $k$-dimensional space (i.e., $\tilde{\mathbf{q}} = \mathbf{U}^{\mathrm{T}}\mathbf{q}$). We use an equivalent formula where we expand documents by their LSI-based dimensions,

$$\tilde{\mathbf{C}} = \lambda \mathbf{C} + (1 - \lambda)\mathbf{V}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}}$$

We then score a document by its cluster-expanded representation.[4]

**Theorem 3** *Cluster-based retrieval in the vector space model is a form of regularization.*

*Proof* Our proof is similar to the proof for document expansion.

$$\begin{aligned}\mathbf{f} &= \tilde{\mathbf{C}}\mathbf{q} \\ &= (\lambda \mathbf{C} + (1 - \lambda)\mathbf{V}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}})\mathbf{q} \\ &= \lambda \mathbf{y} + (1 - \lambda)\mathbf{V}^{\mathrm{T}}[\mathbf{U}^{\mathrm{T}}\mathbf{q}] \\ &= \lambda \mathbf{y} + (1 - \lambda)\mathbf{V}^{\mathrm{T}}\mathbf{y}_c\end{aligned} \tag{23}$$

Because the dimensions (clusters) are representable in the ambient space, we can score them as we do documents; here, we use the $k \times 1$ vector, $\mathbf{y}_c$ to represent these scores. Essentially, the document scores are interpolated with the scores of the clusters. □

[4] In practice, the document representations are only based on the cluster information (i.e., $\lambda = 0$). Our ranking function generalizes classic cluster-based retrieval functions.

### 5.2 Language modeling retrieval

Recall that in Sect. 3.2 we used $L_1$ normalized document vectors to compute similarity. The elements of these vectors are estimates of a term's probability given its frequency in the document and the collection. We refer to the $L_1$ normalized document vector as the document language model, $P(w|\theta_d)$. When treated as a very short document, a query can be also represented as $m$-dimensional language model, $P(w|\theta_Q)$. We can rank documents by the similarity of their models to the query model using a multinomial similarity measure such as cross entropy,

$$\mathbf{y} = (\log \mathbf{C})\mathbf{q} \tag{24}$$

where $\mathbf{q}$ is our initial query model and the log is applied to elements of $\mathbf{C}$ (Rölleke et al. 2006). This is rank equivalent to the likelihood of a document generating the sequence of query terms, $P(Q|\theta_d)$.

In the language modeling framework, pseudo-relevance feedback can be defined in several ways. We focus on the "relevance model" technique (Lavrenko 2004). In this case, the original scores are used to weight each document's contribution to the feedback model, referred to as the "relevance model". The relevance model, $P(w|\theta_R)$, is formally constructed by interpolating the maximum likelihood query model, $P(w|\theta_Q)$, and document models, $P(w|\theta_d)$, weighted by their scores

$$P(w|\theta_R) = \lambda P(w|\theta_Q) + (1-\lambda)\left(\sum_{d \in R} \frac{P(Q|\theta_d)}{\mathcal{Z}} P(w|\theta_d)\right) \tag{25}$$

where, as before, $R$ is the set of top $r$ documents, $\mathcal{Z} = \sum_{d \in R} P(Q|\theta_d)$, and $\lambda$ is a weight between the original query model and the expanded model. In terms of Fig. 7, this means using an $L_1$ normalized version of $\mathbf{y}$. In matrix notation,

$$\tilde{\mathbf{q}} = \lambda \mathbf{q} + \frac{(1-\lambda)}{\|\mathbf{y}\|_1} \mathbf{C}^{\mathrm{T}} \mathbf{y} \tag{26}$$

We then score documents according to Eq. 24.

**Theorem 4** *Relevance models are a form of regularization.*

*Proof* Our proof is based on a similar derivation used in the context of efficient pseudo-relevance feedback (Lavrenko and Allan 2006). Recall that we use $(\log \mathbf{C})\tilde{\mathbf{q}}$ to rank the collection. By rearranging some terms, we can view relevance models from a different perspective,

$$\begin{aligned}
\mathbf{f} &= (\log \mathbf{C})\tilde{\mathbf{q}} \\
&= (\log \mathbf{C})\left(\lambda \mathbf{q} + \frac{(1-\lambda)}{\|\mathbf{y}\|_1} \mathbf{C}^{\mathrm{T}} \mathbf{y}\right) \\
&= \lambda (\log \mathbf{C})\mathbf{q} + \frac{(1-\lambda)}{\|\mathbf{y}\|_1} (\log \mathbf{C})\mathbf{C}^{\mathrm{T}} \mathbf{y} \\
&= \lambda \mathbf{y} + \frac{(1-\lambda)}{\|\mathbf{y}\|_1} \mathbf{A}\mathbf{y}
\end{aligned} \tag{27}$$

where $\mathbf{A}$ is an $n \times n$ affinity matrix based on inter-document cross-entropy. Since the relevance model scores can be computed as a function of inter-document affinity and the

initial scores, this is an instance of score regularization. In fact, iterating the process in Eq. 26 has been shown to improve performance of relevance models and provides an argument for considering the closed form solution in Eq. 15 (Kurland et al. 2005).[5] □

Unfortunately, we cannot reduce document expansion in the language modeling framework to regularization. Document expansion in language modeling refers to adjusting the document language models $P(w|\theta_d)$ given information about neighboring documents (Tao et al. 2006). In this situation, the score function can be written as,

$$\mathbf{f} = \log(\lambda \mathbf{C} + (1 - \lambda)\mathbf{AC})\mathbf{q} \tag{28}$$

Because the logarithm effectively decouples the document expansion from the document scoring, the approach used in the vector space model proof cannot be used here.

The language modeling approach to cluster-based retrieval is conceptually very similar to document expansion (Liu and Croft 2004; Wei and Croft 2006). The distribution $P(z|D)$ represents the distribution of subtopics or aspects in a document; we also have $P(w|z)$ representing language models for each of our subtopics. When we interpolate these models with the maximum likelihood document models, we get a score function similar to Eq. 23,

$$\mathbf{f} = \log\left(\lambda \mathbf{C} + (1 - \lambda)\mathbf{V}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}}\right)\mathbf{q} \tag{29}$$

where $\mathbf{V}$ is the $k \times n$ distribution $P(z|D)$ and $\mathbf{U}$ is the $m \times k$ distribution $P(w|z)$. Like document expansion scores, the logarithm prevents converting cluster-based expansion into a regularization form.

It is worth devoting some time to Kurland and Lee's cluster-based retrieval model (Kurland and Lee 2004). The model is used to perform retrieval in three steps. First, each document is scored according to an expanded document model. Second, an $n \times n$ matrix comparing unexpanded and expanded models is constructed. Finally, each document is scored by the linear interpolation of its original (unexpanded) score and the scores of the nearest expanded documents. To this extent, the model combines regularization and document-expansion retrieval in a language modeling framework. Unfortunately, there do not appear to be experiments demonstrating the effectiveness of each of these steps. Is this model an instance of score regularization? Yes and no. The second interpolation process clearly is an iteration of score regularization. The first score is language model document expansion and therefore not regularization.

Recall that the vector space model allowed fluid mathematical movement from query expansion to regularization to document expansion and finally to cluster-based retrieval. This is not the case for language modeling. Language models have a set of rank-equivalent score functions; we adopt cross entropy in our work. The problem, however, is that measures such as the Kullback–Leibler divergence, cross entropy, and query likelihood all are non-symmetric and therefore not valid inner products. This disrupts the comparison to the vector space model derivations because a smooth transition from regularization (Eq. 27) to document expansion is impossible.

[5] In Sect. 3.2, we adopted the symmetric diffusion kernel to compare distributions. The cross-entropy measure here is asymmetric and therefore cannot be used in our closed form solution. Nevertheless, our iterative solution is not constrained by the symmetry requirement. Furthermore, theoretical results for Laplacians of directed graphs exist and can be applied in our framework (Chung 2004; Zhou et al. 2005).

### 5.3 Laplacian eigenmaps

Score regularization can be viewed as nonparametric function approximation. An alternative method of approximation reconstructs $\mathbf{y}$ with smooth basis functions. When put in this perspective, reconstructing the original function, $\mathbf{y}$, using smooth basis functions indirectly introduces the desired inter-document consistency (Belkin and Niyogi 2003). When Fourier analysis is generalized to the discrete situation of graphs, the eigenvectors of $\mathbf{\Delta}$ provide a set of orthonormal basis functions. We can then construct a smooth approximation of $\mathbf{y}$ using these basis functions. In this situation, our solution is,

$$\mathbf{f}^* = \mathbf{E}\left(\mathbf{E}^{\mathrm{T}}\mathbf{E}\right)^{-1}\mathbf{E}^{\mathrm{T}}\mathbf{y} \tag{30}$$

where $\mathbf{E}$ is a matrix consisting of the $k$ eigenvectors of $\mathbf{\Delta}$ associated with the smallest $k$ eigenvalues. These eigenvectors represent the low frequency harmonics on the graph and therefore result in smooth reconstruction.

**Theorem 5** *Function approximation using harmonic functions of the document graph is a form of regularization.*

*Proof* We can view this process from the perspective of cluster-based retrieval. In the vector space model, Eq. 30 can be rewritten as,

$$\begin{aligned}\mathbf{f}^* &= \mathbf{E}\left(\mathbf{E}^{\mathrm{T}}\mathbf{E}\right)^{-1}\mathbf{E}\mathbf{C}\mathbf{q} \\ &= \left[\mathbf{E}\left(\mathbf{E}^{\mathrm{T}}\mathbf{E}\right)^{-1}\right]\left[\mathbf{E}^{\mathrm{T}}\mathbf{C}\right]\mathbf{q} \\ &= \mathbf{V}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}}\mathbf{q}\end{aligned} \tag{31}$$

where the $k \times m$ matrix $\mathbf{U}^{\mathrm{T}}$ represents the basis as *linear* combinations of document vectors and the $n \times k$ matrix $\mathbf{V}^{\mathrm{T}}$ projects documents into the lower dimensional space. In language model retrieval, Eq. 30 can be rewritten as,

$$\begin{aligned}\mathbf{f}^* &= \mathbf{E}\left(\mathbf{E}^{\mathrm{T}}\mathbf{E}\right)^{-1}\mathbf{E}\log(\mathbf{C})\mathbf{q} \\ &= \left[\mathbf{E}\left(\mathbf{E}^{\mathrm{T}}\mathbf{E}\right)^{-1}\right][\mathbf{E}\log(\mathbf{C})]\mathbf{q} \\ &= \mathbf{V}^{\mathrm{T}}\log(\mathbf{U}^{\mathrm{T}})\mathbf{q}\end{aligned} \tag{32}$$

where the $k \times m$ matrix $\mathbf{U}^{\mathrm{T}}$ represents the eigenfunctions as *geometric* combinations of document vectors.

In both situations, new scores are computed as functions of cluster scores and cluster affinities. Therefore, we claim that basis reconstruction methods are an instance of score regularization. □

### 5.4 Link analysis algorithms

Graph representations often suggest the use discrete metrics such as PageRank to re-weight initial retrieval scores (Brin and Page 1998; Cohn and Hofmann 2000; Kleinberg 1998; Kurland and Lee 2005). These metrics can be thought of as functions from a document to a real value, $g_{\mathbf{W}} : \mathcal{D} \rightarrow \Re$. The function is indexed by the weight matrix $\mathbf{W}$ because these metrics are often dependent only on the graph structure. Let $\mathbf{g}$ be the length-$\tilde{n}$ vector of values of $g$ for our $\tilde{n}$ documents. We will refer to this vector as the *graph structure*

*function*. The values in $\mathbf{g}$ are often combined with those in $\mathbf{y}$ by linear combination (e.g., $\mathbf{f} = \mathbf{y} + \mathbf{g}$) or geometric combination (e.g., $\mathbf{f} = \mathbf{y} \circ \mathbf{g}$).

Many of these methods are instances of the spectral techniques presented in Sect. 5.3 (Ng et al. 2001); specifically, PageRank is the special case where only the top eigenvector is considered (i.e., $\mathbf{g} = \mathbf{E}_1$).

We believe it is very important to ask why the graph represented in $\mathbf{W}$ is being used in retrieval. For regularization, the matrix $\mathbf{W}$ by design enforces inter-document score consistency. For hypertext, the matrix $\mathbf{W}$ (by way of $\mathbf{g}$) provides the stationary distribution of the Markov chain defined by the hypertext graph. This can be a good model of page popularity in the absence of true user visitation data. When better user visitation information is available, though, the model provided by $\mathbf{g}$ is less useful (Richardson et al. 2006). When the graph $\mathbf{W}$ is derived from content-based similarities, what does $\mathbf{g}$ mean? It is unclear that content-derived links can be navigational surrogates; the hypothesis has never been tested. Therefore, applications of graph structure functions to content-based graphs seem weakly justified. We believe that the incorporation of graph structure through regularization, by contrast, has a more solid theoretical motivation.

Because the structure information is lost when computing $\mathbf{g}$ from $\mathbf{W}$, we cannot claim that link analysis algorithms are an instance of regularization.

### 5.5 Spreading activation

When viewed as a diffusion algorithm, our work is also related to the many spreading activation algorithms (Belew 1989; Kwok 1989; Salton and Buckley 1988; Wilkinson and Hingston 1991; Croft et al. 1988) and inference network techniques (Turtle and Croft 1990; Metzler and Croft 2004). In these systems, terms and documents form a bipartite graph. Usually only direct relationships such as authors or sources allow inter-document links. These algorithms operate on functions from nodes to real values, $h : \{\mathcal{D} \cup \mathcal{V}\} \rightarrow \Re$. The domain of the functions includes both documents and terms. The domain of the functions in regularization includes only documents. Clearly spreading activation is not a form of regularization.

However, since regularization is a subset of spreading activation techniques, why should we study it on its own? First, it is not clear that the smoothness objective is appropriate for heterogeneous graphs. Asserting that the scores of a term and a document should be comparable seems tenuous. Second, we believe that our perspective is theoretically attractive because of its ability to bring together several pseudo-relevance feedback techniques under a single framework. Nevertheless, the formal study of heterogeneous nodes in a manner similar to score regularization is a very interesting area of future work.

### 5.6 Relevance propagation

Hypertext collections have inspired several algorithms for spreading content-based scores over the web graph (Qin et al. 2005). These algorithms are equivalent to using a hyperlink-based affinity matrix and iterative regularization. A similar approach for content-based affinity has also been proposed (Savoy 1997). The foundation of these algorithms is at times heuristic, though. We believe that our approach places regularization—whether based on hyperlinks or content affinity—in the context of a mathematical formalism.

### 5.7 Summary

In this Sect. 5, we have studied previous methods exploiting corpus structure from the perspective of score regularization. We present a summary of these results in Table 2.

In the course of our derivations, we have sought to generalize and squint when necessary to show similarities between algorithms. In practice, the implementation of these algorithms differs from what is presented here. We believe these implementation differences explain some performance differences and deserve more detailed analysis.

A variety of graph algorithms exist which use links based on content and hyperlinks. These algorithms often are very subtle variations of each other when analyzed. We hope that our discussion will provide a basis for comparing graph-based and corpus structure algorithms for information retrieval.

Finally, we have restricted our discussion of scoring algorithms to two popular approaches: vector space retrieval and retrieval of language models. Certainly other models exist and deserve similar treatment. This section should provide a perspective not on only analyzing query expansion, regularization, and document expansion in other frameworks but also on developing query expansion, regularization, and document expansion for new frameworks.

## 6 Experiments

Having presented a theoretical context for score regularization, we now turn to empirically evaluating the application of regularization to retrieval scores. We conducted two sets of experiments. The first set of experiments studies the behavior of regularization in detail for four retrieval algorithms: one vector space model algorithm (Okapi), two language modeling algorithms (query likelihood, relevance models), and one structured query algorithm (dependence models); we will abbreviate these okapi, QL, RM, and DM. We present detailed results demonstrating improvements and parameter stability. We will refer to these as the *detailed experiments*. The second set of experiments applies regularization to all automatic runs submitted to the TREC ad hoc retrieval track. These experiments demonstrate the generalizability of regularization.

For all experiments, we will be using queries or topics on a fixed collection with pool-based relevance judgments. These judgments come exclusively from previous TREC experiments and allow for reproducibility of results.

### 6.1 Training

Whenever parameters needed tuning, we performed 10-fold cross-validation. We adopt a Platt's cross-validation evaluation for training and evaluation (Platt 2000). We first randomly partition the queries for a particular collection. 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. Using each of the ten possible training sets of size nine, we generate unique evaluation rankings for each of the topics over all partitions. Evaluation and comparison was performed using the union of these ranked lists.

## 6.2 Detailed experiments

For these detailed experiments, we sought baselines which were strong, in sense of high performance, and realistic, in the sense of not over-fitting. Therefore, we first performed cross-validation to construct baseline retrieval scores. We report the specifics of these experiments in the subsequent sections. We describe our experimental data in Sect. 6.2.1 and baseline algorithms in Sects. 6.2.2–6.2.4. We present parameters for our baseline algorithms in Table 3. We also present trained parameter values (or ranges if they were different across partitions). In Sect. 6.2.5 we discuss the free parameters in regularization and our method for selecting parameter values.

### *6.2.1 Topics*

We performed experiments on two data sets. The first data set, which we will call "trec12", consists of the 150 TREC Ad Hoc topics 51–200. We used only the news collections on Tipster disks 1 and 2 (Harman 1993). The second data set, which we will call "robust", consists of the 250 TREC 2004 Robust topics (Voorhees 2004). We used only the news collections on TREC disks 4 and 5. The robust 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 as the query. The topic title is a short, keyword query associated with each TREC topic. We indexed collections using the Indri retrieval system, the Rainbow stop word list, and Krovetz stemming (Strohman et al. 2004; McCallum 1996; Krovetz 1993).

### *6.2.2 Vector space model scores*

We conducted experiments studying the regularization of vector space model scores (Robertson and Walker 1994). In this approach, documents are represented using a standard tf.idf formula,

**Table 3** Parameter sweep values

Parameter ranges considered in the cross-validation. For each topic set, we present the optimal parameter values selected during training. When these values were not stable across partitions, we present the optimal parameter ranges

| | Range | Optimal | |
|---|---|---|---|
| | | trec12 | robust |
| *Okapi* | | | |
| $b$ | [0.1–1.0; 0.1] | 0.3 | 0.3 |
| $k$ | [0.5–2.5; 0.25] | 1.5–2.0 | 0.75 |
| *Query likelihood* | | | |
| $\mu$ | [500–4000; 500] | 2000 | 1000 |
| *Relevance models* | | | |
| $r$ | {5, 25, 50, 100} | 25–50 | 5–25 |
| $\tilde{m}$ | {5, 10, 25, 50, 75, 100} | 100 | 75–100 |
| $\lambda$ | [0.1–0.7; 0.1] | 0.2 | 0.1–0.2 |
| *Dependence model* | | | |
| $\mu_{\text{text}}$ | [500–4000; 500] | 500–1500 | 3000–4000 |
| $\mu_{\text{window}}$ | [500–4000; 500] | 500–2000 | 500 |

$$\tilde{d}_i = \frac{d_i(k+1)}{d_i + k\left((1-b) + b\left(\frac{l_i}{\|\mathbf{l}\|_1/n}\right)\right)} \tag{33}$$

where $\mathbf{d}$ is a length-$m$ document vector where elements contain the raw term frequency, and the vector $\mathbf{l}$ is the length-$n$ vector of document lengths, $l_i = \|\mathbf{d}_i\|_1$. We then score documents according to the inner product with the query vector, $\langle \tilde{\mathbf{d}}, \mathbf{q} \rangle$.

When computing the similarity between documents, we use an alternate formulation,

$$\tilde{d}_i = d_i \log\left(\frac{(n+0.5) - c_i}{0.5 + c_i}\right) \tag{34}$$

where $\mathbf{c}$ is the length-$m$ document frequency vector. We use this weighting scheme due to its success for topical link detection in the context of Topic Detection and Tracking (TDT) evaluations (Connell et al. 2004). We use the inner product, $\langle \tilde{\mathbf{d}}_i, \tilde{\mathbf{d}}_j \rangle$, to define our affinity matrix.

### 6.2.3 Language model scores

Language model systems provide strong baselines. We use query-likelihood retrieval (Croft and Lafferty 2003) and relevance models (Lavrenko 2004). Both of these algorithms are implemented in Indri (Strohman et al. 2004).

In the retrieval phase, we use Dirichlet smoothing of document vectors,

$$\begin{aligned} \tilde{d}_i &= \frac{d_i + \mu P(w|\theta_C)}{\|\mathbf{d}\|_1 + \mu} \\ &= \frac{d_i + \mu \frac{\|\mathbf{w}_i\|_1}{\mathbf{e}^{\mathrm{T}}\mathbf{C}\mathbf{e}}}{\|\mathbf{d}\|_1 + \mu} \end{aligned} \tag{35}$$

and maximum likelihood query vectors,

$$\tilde{\mathbf{q}} = \frac{\mathbf{q}}{\|\mathbf{q}\|_1} \tag{36}$$

We use cross-entropy to rank documents, $\langle \log(\tilde{\mathbf{d}}), \tilde{\mathbf{q}} \rangle$. We optimize the Dirichlet parameter, $\mu$.

For pseudo-relevance feedback, we take the top $r$ documents from this initial retrieval and build our relevance model using Eq. 25. In practice, we only use the top $\tilde{m}$ terms in the relevance model. We optimize the parameters $r$, $\tilde{m}$, and $\lambda$.

When computing the similarity between documents, we use the diffusion kernel and maximum likelihood document models,

$$\tilde{\mathbf{d}} = \frac{\mathbf{d}}{\|\mathbf{d}\|_1} \tag{37}$$

which we found to be superior to smoothed versions for this task.

### 6.2.4 Dependence model scores

Our final baseline system uses a structured query model which incorporates inter-term dependencies (Metzler and Croft 2005). We present this baseline to demonstrate the

applicability of regularization to non-vector space methods. We use the Indri query language to implement full dependence models with fixed parameters of $(\lambda_T, \lambda_O, \lambda_U) = \{0.8, 0.1, 0.1\}$ as suggested in the literature.

#### 6.2.5 *Regularization parameters*

We performed grid search to train regularization parameters. Parameter values considered are,

| Parameter | Range |
|---|---|
| $\alpha$ | [0.1–0.9; 0.1] |
| $k$ | {5, 10, 25} |
| $t$ | {0.1, 0.25, 0.50, 0.75, 0.90} |

where $t$ is only swept for runs using the diffusion kernel

We normalized all scores to zero mean and unit variance for empirical and theoretical reasons (Belkin et al. 2004; Montague and Aslam 2001). We have found that using alternative score normalization methods performed slightly worse but we do not present those results here.

### 6.3 Regularizing TREC ad hoc retrieval track scores

In addition to our detailed experiments, we were interested in evaluating the generalizability of score regularization to arbitrary initial retrieval algorithms. To this end, we collected the document rankings for all automatic runs submitted to the Ad Hoc Retrieval track for TRECs 3–8, Robust 2003–2005, Terabyte 2004–2005, TRECs 3–4 Spanish, and TRECs 5–6 Chinese (Voorhees and Harman 2001). This constitutes a variety of runs and tasks with varying levels of performance. In all cases, we use the appropriate evaluation corpora, not just the news portions as in the detailed experiments. We also include results for the TREC 14 Enterprise track Entity Retrieval subtask. This subtask deals with the modeling and retrieval of entities mentioned in an enterprise corpus consisting of email and webpages. Although all sites participating in TREC include a score in run submissions, we cannot be confident about the accuracy of the scores. Therefore, inconsistent behavior for some runs may be the result of inaccurate scores.

We ran experiments using the cosine similarity described in Sect. 3. Due to the large number of runs, we fix $k = 25$ and sweep $\alpha$ between 0.05 and 0.95 with a step size of 0.05. Non-English collections received no linguistic processing: tokens were broken on white-space for Spanish and single characters were used for Chinese. Entity similarity is determined by the co-occurrence of entity names in the corpus. The optimal $\alpha$ is selected using 10-fold cross validation optimizing mean average precision.

## 7 Results

In this section, we describe results for our two sets of experiments. Section 7.1 presents a detailed analysis of regularizing several strong baselines. Section 7.2 presents results

demonstrating the generalizability of regularization to scores from arbitrary initial retrieval systems.

### 7.1 Detailed experiments

Our first set of experiments explored the impact of score regularization on four state-of-the-art baselines. We present results for regularizing these scores in Table 4. Results show regularization for different baseline retrievals and different collections. We notice significant improvements across all four algorithms across all collections. This improvement is more pronounced for the techniques which do not use pseudo-relevance feedback (okapi and QL). As noted earlier, our pseudo-relevance feedback run (RM) bears theoretical similarities to regularization (Sect. 5.2) and therefore may not garner rewards seen by other methods. Nevertheless, even this run sees significant gains in mean average precision. Regularizing the dependence model scores produce rankings which out-perform baseline relevance model scores for the robust collection.

Next, we examine the impact of our choice of Laplacian. In Sect. 4.1, we described three alternative definitions of the graph Laplacian. Because our top $\tilde{n}$ documents were likely to be a non-uniform sample across topics, we adopted the approximate Laplace–Beltrami operator which addresses sampling violations. In order to evaluate this choice of Laplacian, we compared the improvements in performance (i.e., map of the regularized scores minus map of the original scores) for all three Laplacians. Our hypothesis was that the approximate Laplace–Beltrami operator, because it is designed to be robust to sampling violations, would result in strong improvements in performance. The results of this comparison are presented in Fig. 8. In all cases the simple combinatorial Laplacian underperforms other Laplacians. Recall from Eq. 7 that, although it weights the comparisons in scores between documents using $W_{ij}$, the combinatorial Laplacian does not normalize this weight by the node degrees (i.e., $D_{ii}$). Both the normalized Laplacian (Eq. 10) and the approximate Laplace–Beltrami operator (Eq. 11) normalize this weight. However, there do not appear to be significant advantages to using the approximate Laplace–Beltrami operator over the normalized Laplacian.

Our first set of experiments, described in Table 4, demonstrated improvements across all four baseline algorithms. The $\alpha$ parameter controls the degree of regularization. In Fig. 9, we plot the effect of regularization as a function of this parameter. Baseline algorithms which did not use pseudo-relevance feedback benefited from more aggressive

**Table 4** Effect of regularization on mean average precision

| | trec12 | | | robust | | |
|---|---|---|---|---|---|---|
| | **y** | **f*** | | **y** | **f*** | |
| okapi | 0.2600 | **0.2834** | +9.02% | 0.2652 | **0.2826** | +6.53% |
| QL | 0.2506 | **0.2778** | +10.86% | 0.2649 | **0.2929** | +10.58% |
| RM | 0.3154 | **0.3252** | +3.12% | 0.2961 | **0.3068** | +3.60% |
| DM | 0.2603 | **0.2833** | +8.84% | 0.2769 | **0.3022** | +9.13% |

This table compares the mean average precision of original scores (**y**) and regularized scores (**f***) for trec12 and robust collections using several baseline scoring algorithms. Bold numbers indicate statistically significant improvements in performance using the Wilcoxon test ($p < 0.05$)

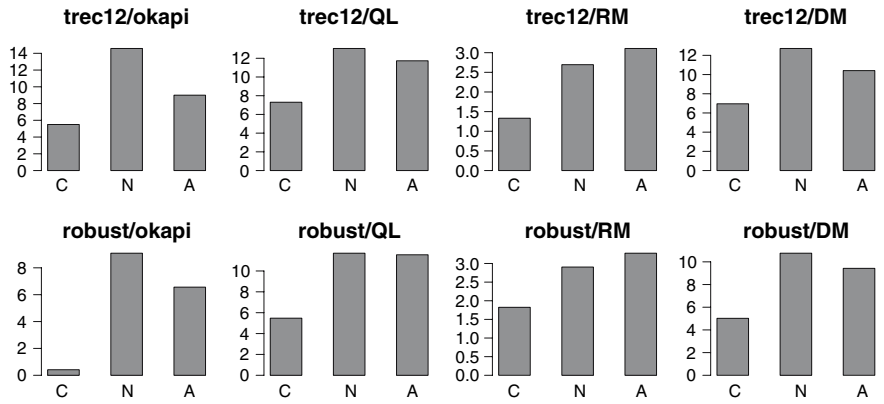


**Fig. 8** Performance improvement as a function of Laplacian type. For each Laplacian described in Sect. 4.1, we maximized mean average precision using 10-fold cross-validation (*left*: combinatorial Laplacian, *center*: normalized Laplacian, *right*: approximate Laplace–Beltrami). The different Laplacians represent different degree normalization techniques

regularization. The pseudo-relevance feedback baseline peaks when initial and regularized scores are more equally weighted.

One of the core assumptions behind our technique is the presence of an underlying manifold or lower-dimensional structure recovered by the graph. The number of neighbors ($k$) represents how much we trust the ambient affinity measure *for this set of documents*. If performance improves as we consider more neighbors, manifold methods seem less justified. In order to test this assumption, we evaluate performance as a function of the number of neighbors in Fig. 10. Across all algorithms and all distance measures, we notice a degradation in performance as more neighbors are considered. This occurs even in the presence of a soft nearest neighbor measure such as the diffusion kernel.

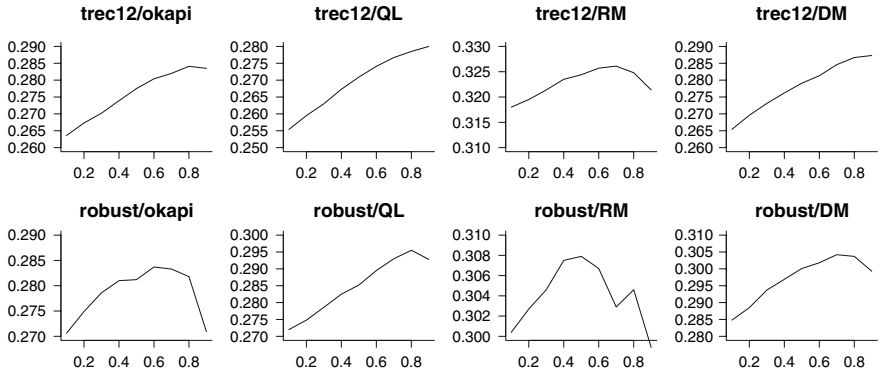


**Fig. 9** Performance as a function of amount of regularization. For each value of $\alpha$, we selected the values for $k$ and $t$ maximizing mean average precision. A higher value for $\alpha$ results in more aggressive regularization. A low value of $\alpha$ recovers the original scores

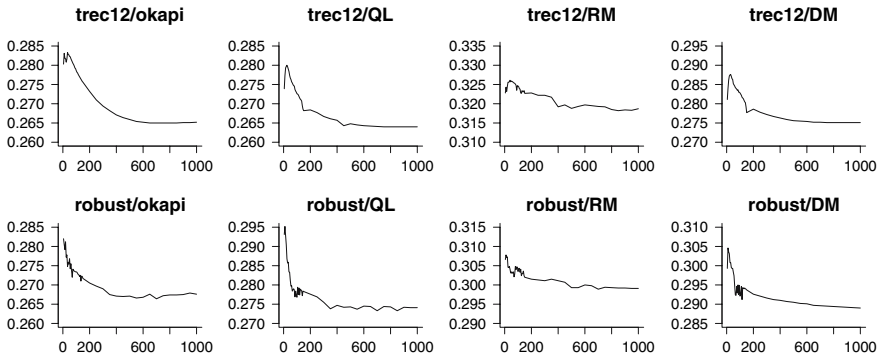


**Fig. 10** Performance as a function of number of neighbors. For each value of $k$, we selected the value for $\alpha$ and $t$ maximizing mean average precision. If we trust the distance metric, we would expect the performance to increase as we increase the number of neighbors

### 7.2 Regularizing TREC ad hoc retrieval track scores

Our detailed experiments demonstrated the improvement of performance achieved by regularizing three strong baselines. We were also interested in the performance over a wide variety of initial retrieval algorithms. We present results for regularizing the TREC Ad Hoc submissions in Figs. 11 and 12 using cosine similarity.[6] Although regularization on average produces improvements, there are a handful of runs for which performance is significantly degraded. This reduction in performance may be the result of an unoptimized $k$ parameter. Improvements are consistent across collections and languages.

## 8 Discussion

We proposed score regularization as a generic post-processing procedure for improving the performance of arbitrary score functions. The results in Figs. 11 and 12 provide evidence that existing retrieval algorithms can benefit from regularization.

We see the benefits in Table 4 when considering several different baselines. However, we can also inspect the improvement in performance as a function of the number of documents being regularized ($\tilde{n}$). In Fig. 13, we notice that performance improves and then plateaus. Though regularization helps both Okapi and QL, the improvement is never comparable to performing pseudo-relevance feedback. This means that despite there being theoretical similarities between regularization and pseudo-relevance feedback, there is a strength in the second retrieval missing in regularization. Nevertheless, our strong single-retrieval algorithm, dependence models, achieves performance comparable to relevance models when regularized.

The results in Figs. 8 and 10 suggest that the construction of the diffusion operator is sometimes important for regularization efficacy. Since there are a variety of methods for constructing affinity and diffusion geometries, we believe that this should inspire a formal study and comparison of various proposals.

The results in Fig. 10 also allow us to test the manifold properties of the initial retrieval. The flatness of the curves for the relevance model run means that the ambient measure

[6] We noticed that the cosine similarity in general outperformed the diffusion kernel.

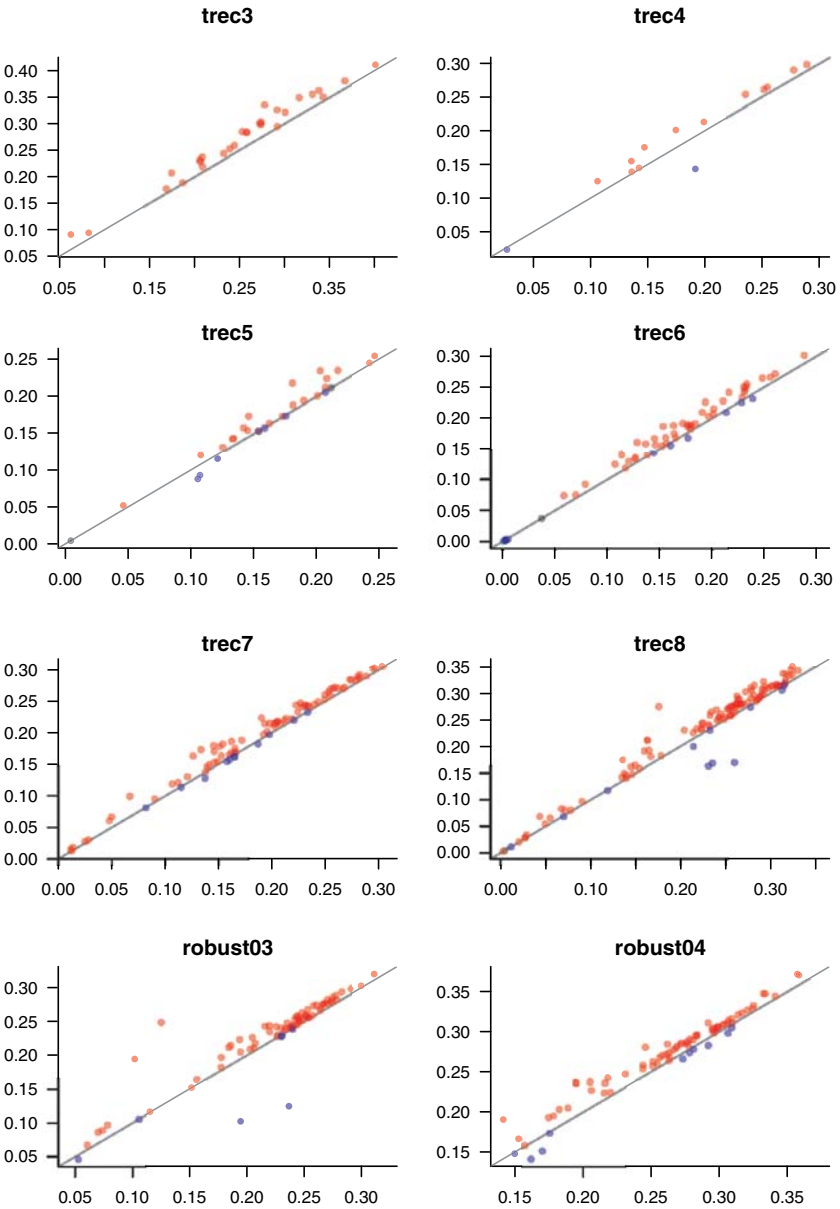


**Fig. 11** Improvement in mean average precision for TREC query-based retrieval tracks. Each point represents a competing run. The horizontal axis indicates the original mean average precision for this run. The vertical axis indicates the mean average precision of the regularization run. Red points indicate an improvement; blue points indicate degradations

behaves well for the documents in this retrieval. Poorer-performing algorithms, by definition, have a mix of relevant and non-relevant documents. Including more edges in the graph by increasing the value of $k$ will be more likely to relate relevant and non-relevant documents. From the perspective of graph-based methods, the initial retrieval for poorer-performing

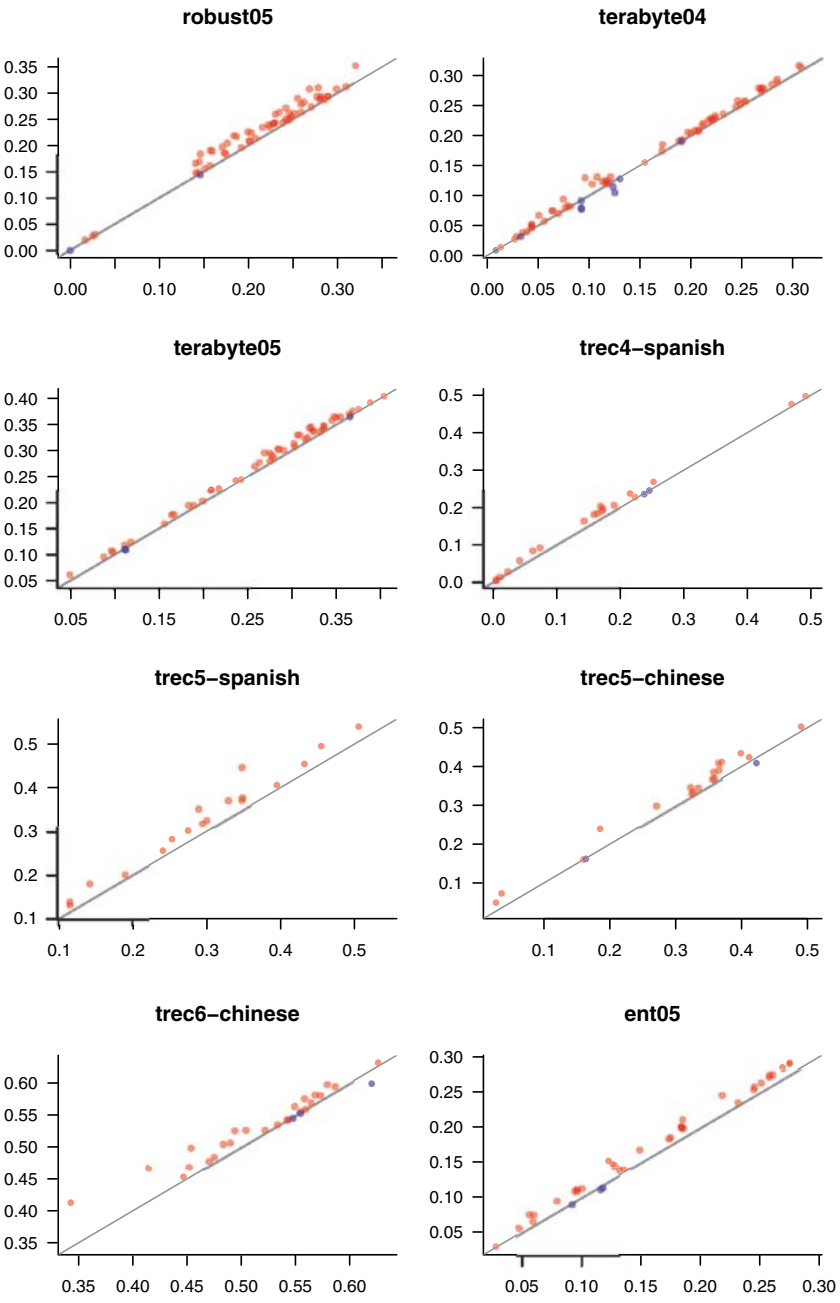


**Fig. 12** Improvement in mean average precision for TREC query-based retrieval tracks. Each point represents a competing run. The horizontal axis indicates the original mean average precision for this run. The vertical axis indicates the mean average precision of the regularization run. Red points indicate an improvement; blue points indicate degradations

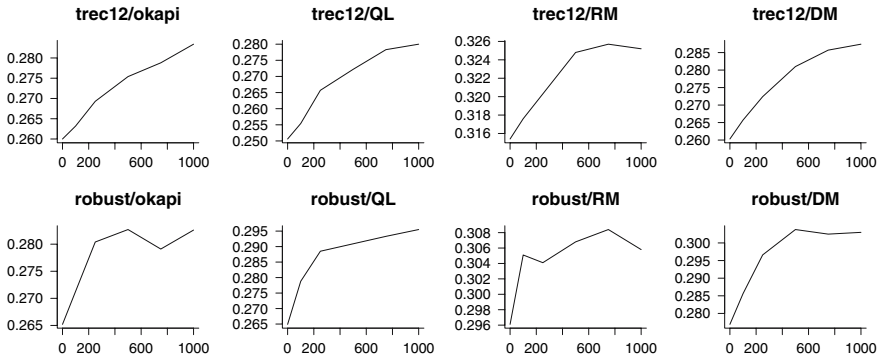


**Fig. 13** Performance as a function of number of documents used for regularization. For each value of $\tilde{n}$, we selected the values for $\alpha$, $k$ and $t$ maximizing mean average precision. A higher value for $\tilde{n}$ considers more documents in the regularization

algorithms should be aggressively sparsified with low values for $k$. On the other hand, better performing algorithms may benefit less from a graph-based representation allowing us to let $k$ grow. From a geometric perspective, documents from poorer-performing algorithms are retrieved from regions of the embedding space so disparate that affinity is poorly-approximated by the ambient affinity. Documents from better performing queries all exist in a region of the embedding space where affinity is well-approximated by the ambient affinity.

We have noted that the aggressiveness of regularization ($\alpha$) is related to the performance of the initial retrieval. Figure 9 demonstrates that smaller values for $\alpha$ are more suitable for better-performing algorithms. This indicates that the use of techniques from precision prediction may help to automatically adjust the $\alpha$ parameter (Carmel et al. 2006; Cronen-Townsend et al. 2002; Yom-Tov et al. 2005).

Finally, we should address the question of efficiency. There are two points of computational overhead in our algorithm. First, the construction of the $\tilde{n} \times \tilde{n}$ affinity matrix requires $O(\tilde{n}^2)$ comparisons. For $\tilde{n} = 1{,}000$, this took approximately 8 s. Although most of our experiments use $\tilde{n} = 1{,}000$, our results in Fig. 13 show that $\tilde{n}$ need not be as large as this to achieve improvements. For example, for $\tilde{n} = 100$, this computation takes less than 0.5 s. We should also point out that we can compute the entire collection affinity matrix and store it prior to any retrieval. In Fig. 10, we showed that only very few neighbors were required to perform optimal performance. This implies that the storage cost would be $O(nk)$. The second point of

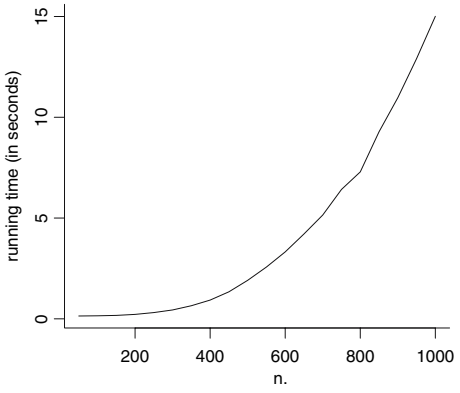


**Fig. 14** Running time as a function of number of documents used for regularization. For each value of $\tilde{n}$, we regularized the scores given a pre-computed affinity matrix

computational overhead is in the inversion of the matrix in Eq. 15. We show running time as a function of $\tilde{n}$ in Fig. 14. Note that our experiments, although very expensive when $\tilde{n} = 1{,}000$, can be computationally improved significantly by reducing $\tilde{n}$ to 500 which, according to Fig. 13, would still boost baseline performance. We could also address the inversion by using the iterative solution. In related work, using a pre-computed similarity matrix and an iterative solution allowed the use of theoretical results from Sect. 5.2 to conduct real-time pseudo-relevance feedback (Lavrenko and Allan 2006).

## 9 Conclusions

We have demonstrated the theoretical as well as the empirical benefits of score regularization. Theoretically, regularization provides a generalization of many classic techniques in information retrieval. By presenting a model-independent vocabulary for these techniques, we believe that the disparate areas for information retrieval can be studied holistically. Empirically, we have shown that regularization can be used as a black box method for improving arbitrary retrieval algorithms. Because of the consistent improvements and potential extensions, we believe that regularization should be applied whenever topical correlation between document scores is anticipated. Furthermore, we believe that, if possible, regularization should be used as a design principle for retrieval models.

**Acknowledgments** This work was supported in part by the Center for Intelligent Information Retrieval and in part by the Defense Advanced Research Projects Agency (DARPA) under contract number HR0011-06-C-0023. Any opinions, findings and conclusions or recommendations expressed in this material are the author's and do not necessarily reflect those of the sponsor. I would like to thank Rosie Jones, Hema Raghavan, Donald Metzler, James Allan, and the anonymous reviewers for providing helpful feedback and Desislava Petkova for providing the Entity Retrieval data. Many of the experiments in this paper would not have been tractable without Andre Gauthier's technical assistance.

## References

Baliński, J., & Daniłowicz, C. (2005). Re-ranking method based on inter-document distances. *Information Processing and Management, 41*(4), 759–775.

Belew, R. K. (1989). 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* (pp. 11–20). New York: ACM Press.

Belkin, M., Matveeva, I., & Niyogi, P. (2004). Regularization and semi-supervised learning on large graphs. In *COLT* (pp. 624–638).

Belkin, M., & Niyogi, P. (2003). Using manifold structure for partially labeled classification. In S. T. S. Becker & K. Obermayer (Eds.), *Advances in neural information processing systems* (Vol. 15, pp. 929–936). Cambridge, MA: MIT Press.

Belkin, M., & Niyogi, P. (2004). Semi-supervised learning on Riemannian manifolds. *Machine Learning, 56*(1–3), 209–239.

Belkin, M., & Niyogi, P. (2005). Towards a theoretical foundation for Laplacian-based manifold methods. In *COLT* (pp. 486–500).

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. In *WWW7: Proceedings of the Seventh International Conference on World Wide Web 7* (pp. 107–117). Amsterdam: Elsevier Science Publishers B. V.

Buckley, C., & Voorhees, E. M. (2000). Evaluating evaluation measure stability. In *SIGIR '00: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 33–40). New York: ACM Press.

Carmel, D., Yom-Tov, E., Darlow, A., & Pelleg, D. (2006). What makes a query difficult? In *SIGIR '06: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 390–397). New York: ACM Press.

Chen, Z., & Haykin, S. (2002). On different facets of regularization theory. *Neural Computation, 14*(12), 2791–2846.

Chung, F. R. K. (1997). *Spectral graph theory*. American Mathematical Society.

Chung, F. (2004). Laplacians and the Cheeger inequality for directed graphs. *Annals of Combinatorics, 9*, 1–19.

Cohn, D. A., & Hofmann, T. (2000). The missing link – a probabilistic model of document content and hypertext connectivity. In *NIPS* (pp. 430–436).

Connell, M., Feng, A., Kumaran, G., Raghavan, H., Shah, C., & Allan, J. (2004). *UMass at TDT 2004*. Technical Report CIIR Technical Report IR-357. Department of Computer Science, University of Massachusetts.

Croft, W. B., & Lafferty, J. (2003) *Language modeling for information retrieval*. Kluwer Academic Publishing.

Croft, W. B., Lucia, T. J., & Cohen, P. R. (1988). Retrieving documents by plausible inference: A preliminary study. In *SIGIR 88: Proceedings of the 11th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 481–494). New York: ACM Press.

Cronen-Townsend, S., Zhou, Y., & Croft, W. B. (2002). Predicting query performance. In *SIGIR '02: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 299–306). New York: ACM Press.

Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., & Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science, 41*(6), 391–407.

Fang, H., Tao, T., & Zhai, C. (2004). A formal study of information retrieval heuristics. In *SIGIR '04: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 49–56). New York: ACM Press.

Fang, H., & Zhai, C. (2005) An exploration of axiomatic approaches to information retrieval. In *SIGIR '05: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 480–487). New York: ACM Press.

Harman, D. K. (1993). The first text retrieval conference (TREC-1) Rockville, MD, U.S.A., 4–6 November, 1992. *Information Processing and Management, 29*(4), 411–414.

Jardine, N., & van Rijsbergen, C. J. (1971). The use of hierarchic clustering in information retrieval. *Information Storage and Retrieval, 7*, 217–240.

Kleinberg, J. M. (1998). Authoritative sources in a hyperlinked environment. In *SODA '98: Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms* (pp. 668–677). Philadelphia: Society for Industrial and Applied Mathematics.

Krovetz, R. (1993). 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* (pp. 191–202). New York: ACM Press.

Kurland, O., & Lee, L. (2004). 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* (pp. 194–201). New York: ACM Press.

Kurland, O., & Lee, L. (2005) 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*.

Kurland, O., Lee, L., & Domshlak, C. (2005). Better than the real thing? Iterative pseudo-query processing using cluster-based language models. In *SIGIR '05: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 19–26). New York: ACM Press.

Kwok, K. L. (1989). 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* (pp. 21–30). New York: ACM Press.

Lafferty, J., & Lebanon, G. (2005). Diffusion kernels on statistical manifolds. *Journal of Machine Learning Research, 6*, 129–163.

Lafon, S. (2004). *Diffusion maps and geometric harmonics*. PhD Thesis, Yale University.

Lavrenko, V. (2004). *A generative theory of relevance*. Ph.D. Thesis, University of Massachusetts.

Lavrenko, V., & Allan, J. (2006). *Real-time query expansion in relevance models*. Technical Report IR-473. Amherst: University of Massachusetts.

Liu, X., & Croft, W. B. (2004). Cluster-based retrieval using language models. In *SIGIR '04: Proceedings of the 27th Annual International Conference on Research and Development in Information Retrieval* (pp. 186–193). New York: ACM Press.

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

Metzler, D., & Croft, W. B. (2004). Combining the language model and inference network approaches to retrieval. *Information Processing and Management, 40*(5), 735–750.

Metzler, D., & Croft, W. B. (2005). A Markov random field model for term dependencies. In *SIGIR '05: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 472–479). New York: ACM Press.

Montague, M., & Aslam, J. A. (2001). Relevance score normalization for metasearch. In *CIKM '01: Proceedings of the Tenth International Conference on Information and Knowledge Management* (pp. 427–433). New York: ACM Press.

Ng, A. Y., Zheng, A. X., & Jordan, M. I. (2001). Stable algorithms for link analysis. In *SIGIR '01: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 258–266). New York: ACM Press.

Petersen, K. B., & Pedersen, M. S. (2005) *The matrix Cookbook*. Version 20051003.

Platt, J. (2000). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In P. J. Bartlett, B. Schölkopf, D. Schuurmans, & A. J. Smola (Eds.), *Advances in large margin classifiers*. MIT Press

Qin, T., Liu, T.-Y., Zhang, X.-D., Chen, Z., & Ma, W.-Y. (2005). 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* (pp. 408–415). New York: ACM Press.

Richardson, M., Prakash, A., & Brill, E. (2006). Beyond PageRank: Machine learning for static ranking. In *WWW '06: Proceedings of the 15th International Conference on World Wide Web* (pp. 707–715). New York: ACM Press.

Robertson, S. E., van Rijsbergen, C. J., & Porter, M. F. (1981). Probabilistic models of indexing and searching. In *SIGIR '80: Proceedings of the 3rd Annual ACM Conference on Research and Development in Information Retrieval* (pp. 35–56). Kent: Butterworth & Co.

Robertson, S. E., & Walker, S. (1994). 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* (pp. 232–241). New York: Springer-Verlag Inc.

Rocchio, J. J. (1971). Relevance feedback in information retrieval. In *The SMART retrieval system: Experiments in automatic document processing* (pp. 313–323). Prentice-Hall Inc.

Rölleke, T., Tsikrika, T., & Kazai, G. (2006). A general matrix framework for modelling information retrieval. *Information Processing and Management, 42*(1), 4–30.

Salton, G. (1968). *Automatic information organization and retrieval*. McGraw Hill Text.

Salton, G., & Buckley, C. (1988). 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* (pp. 147–160). New York: ACM Press.

Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM, 18*(11), 613–620.

Savoy, J. (1997). Ranking schemes in hybrid Boolean systems: A new approach. *Journal of the American Society for Information Science, 48*(3), 235–253.

Singhal, A., & Pereira, F. (1999). Document expansion for speech retrieval. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 34–41). New York: ACM Press.

Strohman, T., Metzler, D., Turtle, H., & Croft, W. B. (2004). Indri: A language model-based search engine for complex queries. In *Proceedings of the International Conference on Intelligence Analysis*.

Tao, T., Wang, X., Mei, Q., & Zhai, C. (2006). Language model information retrieval with document expansion. In *HLT/NAACL 2006* (pp. 407–414).

Turtle, H., & Croft, W. B. (1990). Inference networks for document retrieval. In *SIGIR '90: Proceedings of the 13th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 1–24). New York: ACM Press.

Voorhees, E. (2004). Overview of the TREC 2004 Robust Track. In *Proceedings of the 13th Text REtrieval Conference (TREC 2004)*.

Voorhees, E. M., & Harman, D. K. (Eds.). (2001). *TREC: Experiment and evaluation in information retrieval*. MIT Press.

Wei, X., & Croft, W. B. (2006). LDA-based document models for ad-hoc retrieval. In *SIGIR '06: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 178–185). New York: ACM Press.

Wilkinson, R., & Hingston, P. (1991). 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* (pp. 202–210). New York: ACM Press.

Xu, J., & Croft, W. B. (1999). 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* (pp. 254–261). New York: ACM Press.

Yom-Tov, E., Fine, S., Carmel, D., & Darlow, A. (2005). Learning to estimate query difficulty: Including applications to missing content detection and distributed information retrieval. In *SIGIR '05: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 512–519). New York: ACM Press.

Zhou, D., Schölkopf, B., & Hofmann, T. (2005). Semi-supervised learning on directed graphs. In L. K. Saul, Y. Weiss, & L. Bottou (Eds.), *Advances in neural information processing systems* (Vol. 17, pp. 1633–1640). Cambridge, MA: MIT Press.

Zhou, D., Weston, J., Gretton, A., Bousquet, O., & Schölkopf, B. (2004). Ranking on data manifolds. In L. S. Thrun & B. Schölkopf (Eds.), *Advances in neural information processing systems* (Vol. 16, pp. 169–176). Cambridge, MA: MIT Press.