Employing Document Dependency in Blog Search

Mostafa Keikha[1], Mark James Carman[2] and Fabio Crestani[1]
[1]Faculty of Informatics,University of Lugano, Switzerland
[2]Faculty of IT, Monash University, Australia
mostafa.keikha@usi.ch, mark.carman@monash.edu,
fabio.crestani@usi.ch

March 21, 2012

Abstract

The goal in blog search is to rank blogs according to their recurrent relevance to the topic of the query. State of the art approaches view it as an expert search or resource selection problem. We investigate the effect of content-based similarity between posts on the performance of the retrieval system. We test two different approaches for smoothing (regularizing) relevance scores of posts based on their dependencies. In the first approach, we smooth term distributions describing posts by performing a random walk over a document-term graph in which similar posts are highly connected. In the second, we directly smooth scores for posts using a regularization framework that aims to minimize the discrepancy between scores for similar documents. We then extend these approaches to consider the time interval between the posts in smoothing the scores. The idea is that if two posts are temporally close, they are good sources for smoothing each other's relevance scores. We compare these methods with the state of the art approaches in blog search that employ Language Modeling based resource selection algorithms and fusion-based methods for aggregating post relevance scores. We show performance gains over the baseline techniques which do not take advantage of the relation between posts for smoothing relevance

estimates.

Introduction

User generated content is growing very fast and becoming one of the most important sources of information on the Web. Forums, mailing lists, on-line discussions and social networks like facebook are some of these data resources that have attracted researchers lately. Blogs are one of the main sources of information in this category. Millions of people write about their experiences and express their opinions in blogs everyday.

Given the amount of data being generated by users, there is a pressing need for quality retrieval systems for accessing this content. Because of the specific properties of user generated content and different types of information needs that users may have (Mishne & de Rijke, 2006), classical retrieval methods are not always adequate and new investigation is required. In this paper we investigate the term mismatch problem in blog retrieval and adapt some smoothing methods to decrease its effect on retrieval.

The term mismatch problem in blog retrieval is mainly due to the amount of noise in the blog content and generality of topics provided by users. Blog posts are usually more spontaneous and less formal than classical Information Retrieval (IR) test collections like news wires. Thus spelling mistakes are more common in blog posts, which makes it harder to precisely estimate term probability distributions. The high number of spam blogs and spam comments further intensify this problem. On the other hand, users' information needs in the blogosphere are also different from those of general Web users. Mishne and de Rijke compared blog queries to general Web queries and divided blog queries into two broad categories called context and concept queries (Mishne & de Rijke, 2006). In the context queries, users are looking for Named Entities with special interest in new events. While in the concept queries they are looking for information related to one of their topics of interest. These concept topics are usually general and the user is interested in finding blogs that are mostly related to the topic (Mishne & de Rijke, 2006; MacDonald & Ounis, 2008).

In this paper we focus on the blog distillation task where the goal is to answer concept queries in the blogosphere. Since topics in blog distillation are very general, they are often multifaceted and can be discussed from different perspectives (Elsas et al., 2008). This generality of topics makes the term mismatch problem between documents and topics even more important.

We investigate the effect of content-based similarity between posts on the performance of the retrieval system. We use these similarities as complimentary information to the original query to overcome the term mismatch and topic generality problems. In addition to using a graph-based regularization method to regularize the relevance scores (Diaz, 2005), we propose a general graph-based framework for taking into account the content-based relations between posts. We use this model to smooth the term probability estimation.

Another noteworthy aspect of blog distillation, which differentiates it from other IR tasks, is related to the temporal properties of blogs and queries. Since queries are very general, they can evolve over time, and at each specific time period different aspects of a query can be discussed in the blogosphere. By integrating the temporal similarities between posts into the smoothing models, we explore the effect of time in the smoothing of relevance scores or term probabilities of posts.

The goal of this paper can be summarized in answering the following questions:

- Is content-based similarity between posts useful for distinguishing between relevant and non-relevant blogs?

- Can we use content-based similarity between blog posts to smooth their relevance scores?

- Can we use content-based similarity to better estimate term probability distributions?
- Can we extend smoothing models to take into account temporal relations between posts?

In the rest of the paper, we describe the term mismatch problem in blog search and our motivation for using content similarity to solve it. We then review the state of the art methods in blog retrieval. Specifically we discuss our baselines which are Language Modeling based methods for blog retrieval and fusion-based methods for aggregating post relevance scores. Later we describe the investigated smoothing methods, namely a random walk-based method for smoothing term probabilities, and a regularization framework for smoothing relevance scores. We also discuss extensions of those smoothing methods that take into account temporal information. Finally, experimental results over three different blog data sets are discussed.

Motivations

As stated previously, the goal of blog distillation is to retrieve blogs with recurrent interest in a given topic. The large number of bloggers in the world and the variety of their writing style and vocabulary make this problem challenging, especially since blog posts tend to be less formal and their intention is usually less obvious than classical IR test collections.

Term mismatch and spelling mistakes are common in the blog posts, which make it hard to precisely estimate the term probability distributions. For example, in the Blog08 test collection there are more than 40 million unique terms, more than 60% of which occur only once in the whole collection and more than 99% of them can not be found in a standard dictionary [1]. In comparison the Wall Street Journal (WSJ) collection, a typical newswire

[1]For this analysis we used *New Oxford American Dictionary* in order to check if the term exists in the dictionary or not.

corpus, has 34.3% singleton terms and 67.5% out of dictionary terms (Inches et al., 2010). We see that these numbers are much higher in blogs than in other types of text like news.

At the same time, topics in blog search are usually general and multifaceted. This makes the term mismatch more problematic and causes a wide vocabulary gap between the query and the relevant documents (Elsas et al., 2008).

Because of the generality of topics and the ambiguity in the blogosphere, it is hard to extract relevance information from a post. However, for a specific post we can usually find a set of closely related posts among other blogs. These related posts can be seen as complementary information about the content of the post and can be utilized to create a better representation of the post, overcoming to a certain extent the term mismatch problem.

To test if the similarity between posts can provide more information than the original query for retrieving relevant blogs, we performed some preliminary analysis on relevant and non-relevant blogs extracted from the TREC09 relevance judgments (MacDonald et al., 2009).

Insert Figure 1 here.

First we compared the language model generating probability, as a simple relevance score, between posts from relevant and non-relevant blogs for each query. Figure 1 shows the average post relevance scores from relevant blogs compared to the value of non-relevant blogs for each query. As we can see, they are not distinguishable and for some queries, non-relevant blogs have higher average post relevance scores than relevant ones. This means that the original query might not be enough to distinguish between relevant and non-relevant blogs, and thus we need to use some additional information to improve retrieval. (We note that while the query likelihood of posts is not the only component of the blog retrieval models, and other features like length of the blog or blog cohesiveness also have an effect, it is the main part of all models).

Insert Figure 2 here.

Next we compare the similarity between posts from two relevant blogs against the similarity between posts from a relevant blog and a non-relevant blog. Figure 2 shows the average similarities for each query over the same set of posts analysed in Figure 1. We can see that for all the queries, the average similarity between posts from relevant blogs is higher than the average similarity between posts where one comes from a relevant blog and the other comes from a non-relevant one. Thus, while the average posts score for relevant blogs is not consistently higher across the topics (compared to the non-relevant ones), the posts from relevant blogs do have consistently higher similarity to each other than to the posts from non-relevant blogs. This confirms our intuition that content relations between posts can help us to distinguish between a relevant and a non-relevant blog.

The idea of using the content relations between posts can be seen as similar to query expansion techniques, such as pseudo-relevance feedback. It has been shown, however, that simple query expansion techniques using the top retrieved blogs or top retrieved posts are not effective in blog retrieval and do not improve the performance of the system (Elsas et al., 2008). Thus it remains an open challenge to use this content-relationship information effectively in the blog retrieval context.

Related work

Research in the blog distillation problem started mostly after 2007, when the TREC organizers proposed the task as part of the blog track. Researchers have applied different methods from areas that are similar to the blog distillation such as ad-hoc search, expert search, and resource selection in distributed information retrieval. In the following we describe the main techniques that have been applied to blog retrieval.

In addition we will present a short review on graph-based smoothing techniques in information retrieval and discuss their similarity to our work. Since we also use temporal information to extend the smoothing methods, we will also present works on temporal analysis on blogs. At the end of the section we describe in detail the baselines of our experiments.

Blog Distillation Methods

The simplest models use ad-hoc search methods for finding relevant blogs to a specific topic. They treat each blog as one long document created by concatenating all of its posts together (Efron et al., 2008; Elsas et al., 2008; Seo & Croft, 2008). These methods ignore any specific property of blogs and usually use standard IR techniques to rank blogs. Despite their simplicity, these methods perform fairly well in blog retrieval. In the next section we will discuss one of these methods, the Large Document Model (Elsas et al., 2008), in more detail, as we use it as baseline in our experiments

Some other approaches have been applied to blog retrieval from expert search methods. Expert search is a task in the TREC Enterprise Track where systems are asked to rank candidate experts with respect to their predicted expertise about a query, using documentary evidence of the expertise found in the collection (Soboroff et al., 2007). Some works have shown similarity between blog distillation and expert search, and have adapted expert retrieval methods for blog retrieval (Balog et al., 2008; MacDonald & Ounis, 2008). In these models each post in a blog is seen as evidence that the blog has an interest in the query topic. Balog *et al.* adapt two language modeling approaches of expert finding and show their effectiveness in blog distillation (Balog et al., 2008). MacDonald *et al.* use data fusion models to combine post-based evidence to compute a final relevance score of the blog (MacDonald & Ounis, 2008). Two of the fusion methods applied by MacDonald *et al.* to blog distillation, namely *ExpCombSum* and *ExpCombMNZ*, are used as two of our baselines and are discussed in more detail later.

Other researchers have employed resource selection methods from distributed information retrieval for blog retrieval. In distributed information retrieval the cost of searching all servers for each query is considered prohibitively expensive, so server selection algorithms are used (Hawking & Thomas, 2005). Queries are then routed only to servers that are likely to have many relevant documents for the query. Elsas *et al.* deal with blog distillation as a resource selection problem (Elsas et al., 2008; Arguello et al., 2008). They model each blog as a collection of posts and use a Language Modeling approach to select the best collection. Similar work is described by Seo *et al.*, which they call Pseudo Cluster Selection (Seo & Croft, 2008). The idea is to create topic-based clusters of posts in each blog, and select the blogs which have the most similar clusters to the query. One of these methods, the Small Document Model, is another baseline in our experiments and are discussed in more detail in the next section (Elsas et al., 2008).

Graph-based Smoothing Methods

None of the previously described techniques have investigated the term mismatch problem by using the dependancy between posts. Due to the noisiness of the data and generality of the queries in blog distillation, term mismatch between queries and documents is a challenging issue that needs to be dealt with. Sometimes the original query might not have enough information to retrieve all relevant documents. We use content-based similarities between posts as complementary information to overcome this problem. We employ graph-based methods for capturing the relation between posts and smoothing their relevance scores or their term probabilities.

The similarity between documents has been used for re-ranking an initial ranked list of documents. Kurland and Lee create a graph by connecting a document to its top k similar documents (Kurland & Lee, 2005). They then use the PageRank algorithm (Brin & Page, 1998) to estimate the centrality of each document in the retrieved set of documents and re-rank them based on their centrality score. Diaz proposes an optimization framework for regularizing relevance scores of initially retrieved documents (Diaz, 2005). The model is based on the cluster hypothesis and tries to give similar scores to similar documents. Mei *et al.* use a similar approach and propose a general optimization framework for smoothing language models (Mei et al., 2008). Crestani exploits the term similarity in the term space in order to solve the term-mismatch problem (Crestani, 2000). All of the mentioned methods consider only one type of node in the graph and need a separate similarity function to calculate the similarity between two objects to be used as the weight of the edges in the graph.

Having multiple types of objects in a unified graph (matrix) representation has been considered in different works in information retrieval and recommendation systems. Robertson *et al.* (Robertson et al., 1982) propose a unified probabilistic model that considers both user information needs and documents to calculate the probability of relevance for a given document. Poblete *et al.* (Poblete et al., 2008) propose a unified graph representation for document ranking that includes documents and queries as two types of objects and covers both structural and usage information of the web pages. Similar work has been done by Craswell and Szummer over a click graph of queries and images in order to improve image retrieval perfomance (Craswell & Szummer, 2007). Serdyukov *et al.* model the expert finding task as a random walk in a graph that consists of candidates and documents as its nodes (Serdyukov et al., 2008a, 2008b). Clements *et al.* (Clements et al., 2010) use a tripartite graph that includes users, tags and items where edges show the relation between these nodes. They employ this model for different recommendation and retrieval tasks.

In our models, beside using a regularization framework for smoothing relevance scores of posts, we propose a unified graph that includes both terms and documents as nodes. In this model, smoothed similarities between objects are calculated by a random walk process and there is no need for a separate similarity function.

Temporal Analysis in Blog Retrieval

Further, we extend smoothing models to take into account the temporal similarity between posts. Since the topics are very general and multifaceted, different aspects of the topic might be discussed in the blogosphere. Therefore two similar posts with similar time stamps, might talk about the same aspect of the query and it can be applicable for smoothing their language models.

Time has been used in ad-hoc information retrieval in different ways. Diaz and Jones use time-based query profiles for predicting query precision (Jones & Diaz, 2007). Li and Croft (Li & Croft, 2003) and Erfon and Golovchinsky (Efron & Golovchinsky, 2011) use time-based methods in order to rank recent information for those queries that recency is an important dimension. More related to our problem, temporal properties of posts have been considered in different ways in blog retrieval. Nunes *et al.* use temporal evidence as an extra feature of blogs beside their content (Nunes et al., 2009) . They use the temporal span and the temporal dispersion as two measures of relevance over time, and show that these features can help in blog retrieval. Some models give higher scores to more recent posts before aggregating them. Such models show some improvement over the baseline which uses only the content of the blog (Ernsting et al., 2007; Weerkamp et al., 2008). MacDonald and Ounis try to capture recurring interests of blogs over time (MacDonald & Ounis, 2008). Following the intuition that a relevant blog will continue to publish relevant posts throughout the timescale of the collection, they divide the collection into a series of equal time intervals. They then score blogs according to the number of their relevant posts in different time intervals. Previous works do not look at the temporal distance between posts and its usages in blog retrieval. We employ the temporal distance between two similar posts as a source of smoothing for their language models and integrate it into a blog retrieval method.

Language Modeling based Resource Selection Techniques

Resource selection in distributed information retrieval is a similar problem to the blog distillation task, where the goal is to rank collections of documents. Some models, namely the Small Document Model and the Large Document Model, deal with blog distillation as a resource selection problem (Elsas et al., 2008; Arguello et al., 2008). They model each blog as a collection of posts and use a Language Modeling approach to select the best collection. These methods are well justified from a probabilistic perspective and have been shown to perform well in practice. We have thus chosen to use these methods as baselines for our analysis and therefore describe them in more detail below.

We outline the models presented in (Elsas et al., 2008), where the authors view blog search as the problem of ranking document collections. Following the Language Modeling approach to IR (Zhai & Lafferty, 2004), blogs are ranked according to their likelihood given the query:

$$P(B|Q) = \frac{P(B)P(Q|B)}{P(Q)} \overset{rank}{=} \underbrace{P(B)}_{\substack{\text{Blog}\\ \text{Prior}}} \underbrace{P(Q|B)}_{\substack{\text{Query}\\ \text{Likelihood}}} \qquad (1)$$

If other types of information are available, like statistics regarding the number of subscribers for each blog, they can be used to set the blog prior $P(B)$ to an appropriate value. Otherwise the blog prior can be set uniformly or allowed to grow logarithmically with the number of posts in the blog, so as to favor longer blogs since they are more likely to contain useful information (Elsas et al., 2008).

Once a blog prior is chosen, we need to estimate the query likelihood. One of the methods proposed in (Elsas et al., 2008), the Large Document Model (LDM), treats each blog as a single document that results from concatenating all of its posts together. The query likelihood $P(Q|B)$ in this case is estimated using Dirichlet-smoothed term probability estimates as follows:

$$P_{\texttt{LDM}}(Q|B) = \prod_{t \in Q} P_{\texttt{LDM}}(t|B) = \prod_{t \in Q} \left(\frac{\mathrm{tf}(t,B) + \mu P_{\texttt{ML}}(t|C)}{(\sum_t \mathrm{tf}(t,B)) + \mu} \right) \tag{2}$$

where $\mathrm{tf}(t,B)$ is the total number of occurrences of term t in posts in the blog, μ is a smoothing parameter, and $P_{\texttt{ML}}(t|C)$ is the Maximum Likelihood (ML) estimate for the term distribution for the collection as a whole (i.e., the relative frequency across all the blogs).

An alternative model, the Small Document Model (SDM), treats each post within a blog as a separate document and attempts to quantify the importance (centrality) of each post to the rest of the content in the blog. The query likelihood for a blog is calculated by summing the query likelihoods for each post in the blog scaled according to the probability (centrality) of the post within the blog:

$$P_{\texttt{SDM}}(Q|B) = \sum_{p \in B} P(Q|p)P(p|B) \tag{3}$$

Here p is the post in the blog, and $P(Q|p)$ is the query likelihood for each post which is computed over query terms using Jelinek-Mercer smoothing:

$$P(Q|p) = \prod_{t \in Q} \lambda_p P_{\texttt{ML}}(t|p) + \lambda_B P_{\texttt{ML}}(t|B) + \lambda_C P_{\texttt{ML}}(t|C) \tag{4}$$

Where $\sum \lambda_* = 1$, $\lambda_* \geq 0$ and $P_{\texttt{ML}}(t|*) = \frac{tf(t,*)}{\sum_t tf(t,*)}$. The post centrality is computed using likelihood that the blog would generate the terms in the post:

$$P(p|B) \approx \prod_{t \in p} P(t|B)^{P_{\mathtt{ML}}(t|p)} \tag{5}$$

Here $P(t|B)$ is estimated by the average of the probabilities of the term in the blog posts using $\frac{1}{N_B} \sum_{p \in B} P_{\mathtt{ML}}(t|p)$ where N_B is the number of posts in the blog (Elsas et al., 2008).

Rather than using these complex methods, our experiments show that uniform estimations perform well and are easier to calculate. Thus, beside using described estimations, we also use uniform estimations, which we denote by $SDM - uniform$, in our experiments. For this ranking formula we consider a uniform prior over the blogs and as the post centrality use a uniform distribution over posts in each blog. The query likelihood for a blog is then calculated by:

$$P_{\mathtt{SDM-uniform}}(Q|B) = \frac{1}{N_B} \sum_{p \in B} P(Q|p) \tag{6}$$

Between two blogs with the same $\sum_{p \in B} P(Q|p)$, the method gives higher score to the blog with the smaller number of posts because the query topic is more likely to be the main topic in the smaller blog. This is similar to the estimations used by Balog *et al.* (Balog et al., 2008).

Aggregation Techniques from Expert Search

Expert Search is a task in the TREC Enterprise Track where systems are asked to rank candidate experts with respect to their predicted expertise about a query, using documentary evidence of their expertise found in the collection (Soboroff et al., 2007). In (MacDonald & Ounis, 2006b), the authors create a voting model for expert search and later use this model also for blog search (Hannah et al., 2008). In this model, bloggers are seen as experts in different topics. A blogger with an interest in a particular topic will oftentimes blog regularly about the topic, and his/her posts will likely be retrieved in response to a query about it. Blog distillation can then be seen as a voting process: each time a post is retrieved in response to a query, it is considered as a weighted vote for the expertise of the blogger in that particular topic. They then use fusion methods to aggregate the relevance scores of posts in each blog and rank the blogs based on the aggregated scores. The best performing aggregation methods in their experiments were *ExpCombSum* and *ExpCombMNZ*. The first method *ExpCombSum* ranks blogs according to the sum of the exponent of the relevance scores for most relevant posts from the blog:

$$score_{\texttt{ExpCombSum}}(B, Q) \;=\; \sum_{p \in B \cap R(Q)} \exp(score(p, Q)) \tag{7}$$

Here $R(Q)$ denotes the set of posts retrieved for query Q, the intersection $B \cap R(B)$ denotes only those retrieved posts that come from blog B and $score(p, Q)$ denotes the relevance score assigned to the post p for the query by the underlying search engine. The second method *ExpCombMNZ* takes into account number of the retrieved posts from each blog in the rankin:

$$score_{\texttt{ExpCombMNZ}}(B, Q) \;= |B \cap R(Q)| \sum_{p \in B \cap R(Q)} \exp(score(p, Q)) \tag{8}$$

We use *ExpCombSum* and *ExpCombMNZ* as other baselines in our comparisons. Other features like cohesiveness (how similar are the posts in a blog to each other) and anchor text similarity to the query (from links pointing to the blog) are also investigated in blog search. However, experiments indicated that these features did not improve performance significantly (Hannah et al., 2008).

It is worth noting that all the blog retrieval methods need an aggregation step in order to calculate the blog relevance based on its posts relevance. It can be seen similar to the aggregation techniques employed in XML retrieval where the depth of the XML tree is limited to 2 (Ogilvie & Callan, 2004).

Document Dependency in the Blogosphere

None of the techniques described in previous section have taken the dependency between posts from different blogs into account when calculating query relevance scores for blogs. In this paper we investigate whether the dependency between posts both within and across blogs can be used to improve blog retrieval performance. First we consider the content-based relation to smooth document language models. Then we extend it by considering the temporal-closeness of posts as another relation and employing it in smoothing.

Smoothing Term Probabilities using Random Walks

One way to make use of the relationships between posts is to smooth the term probability distributions for each post by taking into account the number of terms it has in common with other posts. The more common terms there are between two posts, the more similar their term probability distributions should be.

We propose a graph based representation of the most relevant posts (for a particular query) and their terms. Figure 3 shows part of such a graph where an edge between a post and a term indicates that the term occurred in that specific post.

Insert Figure 3 here

We record this relation in a transition matrix A. In this matrix A_{ij} (element of row

i and column j in the matrix A) denotes the transition probability from node i to node j in one step of a random walk, i.e. $P(t_j|p_i) = A_{ij}$ shows the probability of a term for a given post (transition probability from a post to a term) and $P(p_i|t_j) = A_{ji}$ shows the probability of a post for a given term (transition probability from a term to a post). The outgoing probabilities from a node and hence the values in any row of A add up to one, i.e. $\sum_j A_{ij} = 1$. It is worth noting that the matrix A is a square matrix in which number of rows (columns) is equal to the total number of posts and terms.

We will use the notation $P_n(t_j|p_i) = (A^n)_{ij}$ to denote the transition probability from post p_i to query term t_j in n steps. Performing this random walk can be seen as a type of smoothing where we compute probability estimates even for terms not present in the document. This graph-based smoothing takes into account the frequency of each term in similar documents, where similar documents are determined by their common terms. For example, even if a particular blog post discussing some aspect of machine learning did not contain the word "regularization", the term would be assigned a non-zero value within a smoothed term probability distribution for the post, since other posts discussing machine learning would likely contain this term with high frequency.

Importantly, whenever a term occurs in a smoothed document model, the random walk can explain its presence and the probability value assigned. An obvious benefit of this approach is that we do not need to calculate similarity between all pairs of posts as done in the regularization framework, since the similarity is calculated implicitly during the random walk process.

Since we only need to compute $P_n(t_j|p_i)$ for terms in the query, we can efficiently calculate this value, as indicated in (Craswell & Szummer, 2007), by iterating forward from *each* query term node t_j to *all* post nodes p_i concurrently, as follows:

$$P_n(t_j|p_i) = (A^n)_{ij} = (A(...(A(A\mathbf{j})))_i \tag{9}$$

Where **j** is a unit (column) vector with value 1 in the j-th row. Note that this calculation only needs to be performed once per query term.The transition probabilities between terms and documents are calculated using Maximum Likelihood estimate:

$$P(t_j|p_i) = \frac{\mathrm{tf}(t_j, p_i)}{|p_i|} = \frac{\mathrm{tf}(t_j, p_i)}{\sum_k \mathrm{tf}(t_k, p_i)} \tag{10}$$

Where $\mathrm{tf}(t_j, p_i)$ is the frequency of the term t_j in the post p_i and $|p_i|$ shows the size of the post which is the total number of terms in it. Similarly, the probability of transitioning from a term node t_j to a post node p_i is defined using Bayesian theorem by:

$$P(p_i|t_j) = \frac{\mathrm{tf}(t_j, p_i)}{\sum_k \mathrm{tf}(t_j, p_k)} \tag{11}$$

This formula can be concluded directly from equation 10 by assigning proper prior probabilities for posts and terms. We set the prior probability of a post to be proportional to its size:

$$p(p_i) = \frac{|p_i|}{\sum_k |p_k|} \tag{12}$$

and probability of a term is set to be proportional to its frequency in the collection:

$$p(t_j) = \frac{\sum_k \mathrm{tf}(t_j, p_k)}{\sum_k |p_k|} \tag{13}$$

Using these prior probabilities in Bayesian theorem equations 10 and 11 can be derived from each other.

A random walk of length one is equivalent to a Maximum Likelihood estimate for $P_n(t_j|p_i)$, while an infinite random walk would generate a stationary probability distribution independent of the starting points.

By adding a self-loop transition to all term nodes, we turn a length n random walk into the weighted (exponentially decaying) average of walks of length 1 to n. We do this by setting $\alpha = P(t_j|t_j)$ as the "self-loop" probability on the term nodes. This is a smoothing parameter which regulates the importance of shorter versus longer walks in the post-term graph. Thus the parameter α regulates how much smoothing we do on the initial Maximum Likelihood estimate. The smaller the self-loop probability α, the more the estimate will be smoothed with longer walks and vice versa.

To see the effect of α in the final probabilities in the random walk, assume we calculate a walk of length 20 in a graph without self-loop. At the end of this calculation, we have $P_{20}(t_j|p_i)$ for all terms and posts. However the calculated probabilities do not have any information about probabilities in shorter walks and we miss that information. Thus, for example it is possible that the lengths of all paths between a post and a term are odd, so P_{20} for that post and term will be zero, while a shorter path of length 19 might have a non-zero value. By adding the self-loop on the terms, we keep a history of all walks to that term.

Finally, to make the matrix stochastic (where each row sums to one) given the term self-loops, we decrease the probabilities from terms to posts by a factor of $(1 - \alpha)$. The generated matrix looks as follows:

$$A = \begin{bmatrix} 0 & M_{PT} \\ (1-\alpha)M_{TP} & \alpha I \end{bmatrix} \tag{14}$$

Here P indicates the posts, T the terms, and M_{XY} is a stochastic sub-matrix with transition probabilities from the object type X to the object type Y. Once we have computed $P_n(t_j|p_i)$ for each term in the query we can use these values (after further smoothing with a collection model $P(t_j|C)$) to calculate an estimate for the query likelihood given the post:

$$P_{RW}(Q|p_i) = \prod_{t_j \in Q} \lambda P_n(t_j|p_i) + (1-\lambda)P(t_j|C) \tag{15}$$

We note that the random walk based framework for smoothing term distributions is very general and can easily be extended with new forms of evidence such as hyperlinks between posts which can be encoded directly as additional edges into the post-term graph.

Blog Post Scores Regularization

Score regularization is a way to re-calibrate relevance scores of documents based on the relationship between them. The idea behind score regularization is that in accordance with the Clustering Hypothesis, *related documents should have similar scores for the same query.* In (Diaz, 2005; Mei et al., 2008) general models for smoothing document scores were proposed based on this hypothesis. Diaz models the problem in terms of a regularized optimization problem on a document similarity graph (Diaz, 2005). The goal is to calculate for each document a new (smoothed) score with two contending objectives: score consistency with related documents and score consistency with the initial retrieval score. He defines an overall cost function $\zeta(f)$ as follows:

$$\begin{aligned}\zeta(f) &= \sigma(f) + \mu\varepsilon(f) \\ &= \sum_{i \neq j}(w_{ij}f_i - w_{ji}f_j)^2 + \mu \sum_i (f_i - y_i)^2\end{aligned} \tag{16}$$

Where f is a vector of regularized scores over n documents, $\sigma(f)$ is a cost function associated with the inter-document consistency of the scores. High values of $\sigma(f)$ indicate inconsistent scores between related documents. A second cost function $\varepsilon(f)$ measures the consistency with the initial scores; if document scores are inconsistent with the initial scores, the value of this function will be high. A regularization parameter μ controls the trade off between inter-document smoothing and consistency with the initial score vector y. The coefficient w_{ij} in the expansion of $\sigma(f)$ weights the score of the ith document by its similarity to the jth document and is calculated by normalizing (and taking the square root of) values from a symmetric affinity matrix W as follows:

$$w_{ij} = \sqrt{\frac{W_{ij}}{\sum_j W_{ij}}}. \tag{17}$$

Here W_{ij} denotes the similarity between documents i and j. In order to keep the affinity matrix sparse, only the k most similar documents j for each document i have non-zero W_{ij} values[2]. The diagonal values in the matrix W are defined to be zero. An iterative solution for the above optimization problem is the following:

$$f^{t+1} = (1-\alpha)y + \alpha\bar{W}f^t \tag{18}$$

Where $\alpha = 1/(1+\mu)$ is a parameter, $y = f^0$ is the initial score vector, f^t is the score vector after t iterations and $\bar{W}$ is a normalized affinity matrix such that $\bar{W}_{ij} = w_{ij}w_{ji}$. The closed form solution of this problem is given by:

$$f^* = (I - \alpha\bar{W})^{-1}y \tag{19}$$

We will use this equation in our experiments where we apply graph-based regularization frameworks (Diaz, 2005; Mei et al., 2008) to the problem of blog distillation.

[2]Although some documents may need to have more than k non-zero affinity values in order to keep the matrix symmetric.

Smoothing with temporal relations

In another approach of smoothing, we tried to use the temporal relation between posts for smoothing their scores. As mentioned before, a number of works have considered time as an additional feature in blog retrieval demonstrating the importance of time in the blogosphere (Efron et al., 2008; Ernsting et al., 2007; Weerkamp et al., 2008). Also other works have shown a relationship between blogs and news (Sun et al., 2008; Mishne & de Rijke, 2006). They show that some news-related topics are bursty in the blogosphere and are discussed more in specific time intervals. Based on this observation we consider query-related posts with less temporal distance as more dependent than posts that are temporally far away from each other. We now integrate these dependencies as an another source of smoothing in our models.

Insert Figure 4 here.

We define the temporal similarity measure as a decay function:

$$Sim_{temporal}(p_i, p_j) = e^{-\frac{\Delta d}{\sigma}} \tag{20}$$

where Δd is the temporal distance in days and $\frac{1}{\sigma}$ is the decay constant[3]. The temporal transition probabilities are:

$$P_{temporal}(p_j|p_i) = \frac{Sim_{temporal}(p_i, p_j)}{\sum_k Sim_{temporal}(p_i, p_k)} \tag{21}$$

To add the temporal relations to the post-term graph in the random walk model, we add new edges between each pair of posts based on their temporal distance. Figure 4 shows

[3]We tested other decay functions for temporal similarity such as the Gaussian kernel but experiments showed a simple exponential decay to be the best temporal similarity function.

the format of the new graph. We assign the weight of new edge as:

$$P(p_j|p_i) = \beta \ P_{temporal}(p_j|p_i) \tag{22}$$

where $P_{temporal}(p_j|p_i)$ is calculated by equation 21 and $\beta < 1$. We change the weight from posts to terms as follow, to keep the weights of outlinks from each post summing to 1:

$$P(t_j|p_i) = (1-\beta) \ \frac{\mathrm{tf}(t_j, p_i)}{\sum_j \mathrm{tf}(t_j, p_i)} \tag{23}$$

The newly generated matrix looks as follows:

$$A = \begin{bmatrix} \beta \ M_{PP} & (1-\beta) \ M_{PT} \\ (1-\alpha) \ M_{TP} & \alpha \ I \end{bmatrix} \tag{24}$$

We note that there is a new sub-matrix M_{PP} with respect to equation 14 with transition probabilities from posts to posts as described above.

We also add the temporal relations to the regularization framework by changing the affinity function used in creating the affinity matrix. The new similarity measure is:

$$Sim(p_i, p_j) = Sim_{temporal}(p_i, p_j) \ Sim_{content}(p_i, p_j) \tag{25}$$

We use this new similarity as W_{ij} in the affinity matrix and perform regularization of post scores as before.

Computational Complexity

An important issue when comparing the two smoothing methods described above is their scaling behavior and whether their complexity can be reduced through approximation.

For regularization, the algorithm must first compute a kNN affinity matrix and then perform a matrix inversion operation, which results in a complexity of $O(n^2v + n^3)$, where n is the number of documents and v is the average vocabulary of the documents. If we approximate the affinity matrix using a hierarchical clustering approach (Pfahringer et al., 2007) and use the iterative method rather than the closed form to calculate a solution, then the complexity of the algorithm reduces to $O(n \log(n)v + skn)$, where s is the number of iterations, and k is the number of non-zero elements per document in the affinity matrix. Multiplying a sparse matrix by a vector takes $O(m)$ time, where m is the number of non-zero elements in the matrix, which in this case is kn.

For the random walk computation, we need to multiply a sparse matrix by a vector l times, where l is the length of the walk and repeat that process for each term in the query (q times), resulting in a complexity of $O(qlvn)$, since there will be vn non-zero elements in the transition matrix.

Thus we see that in terms of complexity the random walk and the regularization algorithms are almost equally efficient compared to one another depending on the length of the query, the average vocabulary of posts, the length of the random walk, and number of the regularization steps required for convergence.

Experimental Results

To evaluate our methods we use three years worth of TREC blog track data from the blog distillation task, including TREC'07, TREC'08 and TREC'09 data sets. The TREC'07 and TREC'08 data sets include 45 and 50 assessed queries respectively and use Blog06 collection. The TREC'09 data set includes Blog08, a new collection of blogs, and has 39 new queries (MacDonald & Ounis, 2006a; MacDonald et al., 2009). We use just the title of the queries in our experiments.

The Blogs06 collection is a crawl of about one hundred thousand blogs over an 11-

week period (MacDonald & Ounis, 2006a), and includes blog posts (permalinks), the feed, and the homepage for each blog. Blog08 is a collection of about 1 million blogs crawled over a year with the same structure as Blog06 collection (MacDonald et al., 2009). In our experiments we only use the permalinks component of the collection, which consists of approximately 3.2 million documents for Blog06 and about 28.4 million documents for Blog08.

After smoothing the term probabilities using a random walk and regularizing post scores in a regularization framework, we use these new scores in the small document model for calculating blog relevance scores.

Parameter Settings

We use the Terrier Information Retrieval system[4] to index the collection and retrieve documents. Previous experiments show that working just with the most relevant posts (according to a retrieval score) improves the performance and requires less computational effort (Lee et al., 2008). For this reason, we selected the top 2000 posts for each query using the query likelihood probability presented in equation 4. Since we have three years worth of data, we considered each year as a separate test collection. For the parameters of the models, we learnt the values for each year using the other two years as training data and tuned the parameters to optimize each evaluation measure separately. This included tuning λ in equations 4 and 15 for language model smoothing, α in equation 19 for regularization and α and β in equations 14 and 24 for term self-loop and temporal similarity importance. For all these parameters we search for values between 0 and 1 at intervals of 0.1.

For the regularization and random walk based smoothing methods, we create the corresponding affinity and transition matrices using the top 2000 retrieved posts, and after smoothing/regularization, we calculate blog scores by the mentioned methods (SDM and

[4]http://ir.dcs.gla.ac.uk/terrier/

SDM-uniform).

In order to generate a kNN affinity matrix for the regularization method, we use the cosine similarity as a simple and effective similarity metric, that has been shown to be successful in graph based Language Model smoothing (Mei et al., 2008):

$$sim(p_1, p_2) = \frac{\sum_t \mathrm{tf}(t, p_1)\ \mathrm{tf}(t, p_2)}{\sqrt{\sum_t \mathrm{tf}(t, p_1)^2}\sqrt{\sum_t \mathrm{tf}(t, p_2)^2}} \tag{26}$$

For each post, we select the average of its similarity with the other posts as the similarity threshold. We add to the matrix all posts with similarities higher than the threshold and for the rest we put zero.

Before generating the transition matrix for the random walk-based smoothing, we discarded terms with very high document frequency (more than 80% of the documents) or very low document frequency (less than 5 documents) in order to reduce the size of the graph. As with most of the parameters, we set the term self-loop probability α by learning from the training data. The length of the random walk is set to be long enough to approach the stationary distribution, our experiment showed that length of 20 was sufficient for all the generated graphs.

Results

Tables 1, 2, and 3 show the results for the different methods on the TREC'07, TREC'08 and TREC'09 data sets respectively. In order to test for statistical significance, we use the Student's Paired T-test on scores for each query at the 0.05 level. Statistically significant improvements over LDM and SDM-uniform, as the two best baselines, are shown by † and ‡ respectively. We also show the statistically significance improvement of a temporal model, RW-Temporal and Reg-Temporal, over the corresponding non-temporal model by $\star$.

Insert Table 1 here.

Insert Table 2 here.

Insert Table 3 here.

We can see that smoothing methods outperform the baseline in MAP across different data sets. In most of the cases this improvement is statistically significant. For Precision at 10, there are improvements in some cases but the improvement is not consistent across all datasets. It is interesting to see, however, that Precision at 10 is the metric that is affected by the addition of temporal information. In all cases, adding the temporal information improves the Precision at 10 over the non-temporal methods and in some cases the improvement is statistically significant. For the Bpref measure, the two random walk methods perform better than the regularization methods and adding the temporal information improves this measure slightly.

It is interesting to see that SDM-uniform performs the best among all methods without smoothing. This supports the intuition for prior probabilities in this model i.e. with the same relevance evidence, the shorter of two blogs is more likely to be relevant.

In the second part of our experiments we investigate temporal relations between posts for smoothing the scores. In the calculation of temporal similarities between posts, equation 20, we use the average temporal distance between the retrieved posts for each query as the σ value ($1/\sigma$ is the decay constant). As we can see, adding the temporal similarity in the smoothing methods improves the performance of the systems. This means that posts that occur within a shorter time interval from one another are better sources of information for smoothing language models than posts that occur far from each other in time.

Insert Table 4 here.

Figures 5, 6, and 7 show interpolated Precision versus Recall graphs for the three data sets. The graphs compare the two similarity-based smoothing techniques (RW-Temporal and Reg-Temporal) with the best of the Language Modeling techniques (SDM-Uniform). Again we see consistent improvements over the best baseline method (SDM-Uniform) across different levels of Recall for different data sets.

Insert Figure 5 here.

Insert Figure 6 here.

Insert Figure 7 here.

Comparing the random walk (RW) and regularization (Reg) techniques against each other, we see that the two techniques are very much comparable in terms of overall performance. Regularization appears to slightly outperform the random walk technique in terms of MAP, but the situation is reversed for Bpref. Since the difference in the performance of the two techniques is not large we recommend that the choice between the techniques be made based on the computational complexity issues previously mentioned, where factors like the average vocabulary of posts and queries need to be taken into account.

Finally, one important issue for graph-based smoothing is the proportion of relevant and non-relevant documents in the graph. If the percentage of relevant documents in the graph is very low, the result of the smoothing might go toward the non-relevant documents and score them higher than the relevant ones. To test if this was the case in our experiments, we calculated the Kendall τ correlation between the improvement for each query and the number of relevant blogs for that query amongst the relevance judgments. Table 4 shows the correlation coefficients for Reg-Temporal and RW-Temporal over SDM-uniform for the different data sets. We see that in all cases the values are close to zero and none of them is statistically significant. The low correlation coefficients indicate that the performance improvements are not correlated with the number of relevant blogs for each query. Thus, the graph-based smoothing can be effective even when the number of relevant blogs is low.

Conclusion

In this paper we investigated whether the similarity between posts both in and across blogs could be used to smooth post relevance scores and thereby improve blog rankings. We tested two different approaches for smoothing relevance scores, one based on a regularization framework that aims to minimize the discrepancy between scores for similar documents and another based on a random walk algorithm that smoothes the term distributions used to represent posts. We compared the methods with state of the art approaches in blog search that employ Language Modeling based resource selection algorithms and fusion-based methods for aggregating post relevance scores. We found that the smoothing methods performed better than the baseline techniques. There was, however, no significant differences in performance between the two smoothing techniques across the test sets.

We considered the temporal distance between posts as an another source of smoothing and showed that it can improve the performance of the system.

Future work will be directed toward considering other information, such as hyperlinks and user comments, as relations between the posts and to investigate the effect of these relationships on relevance scores. Considering the relationship between blogs and other types of data, like query logs and news articles, may also be effective for improving retrieval performance.

1 References

Arguello, J., Elsas, J., Callan, J., & Carbonell, J. (2008). Document representation and query expansion models for blog recommendation. *Proceedings of ICWSM 2008*.

Balog, K., de Rijke, M., & Weerkamp, W. (2008). Bloggers as experts: feed distillation using expert retrieval models. *Proceedings of SIGIR 2008* (pp. 753–754).

Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, *30*(1-7), 107–117.

Clements, M., de Vries, A. P., & Reinders, M. J. T. (2010). The task-dependent effect of tags and ratings on social media access. *ACM Transaction on Information Systems*, *28*(4), 21.

Craswell, N., & Szummer, M. (2007). Random walks on the click graph. *Proceedings of SIGIR 2007* (pp. 239–246). ACM.

Crestani, F. (2000). Exploiting the similarity of non-matching terms at retrieval time. *Information Retrieval*, *2*(1), 23–43.

Diaz, F. (2005). Regularizing ad hoc retrieval scores. *Proceedings of CIKM 2005* (pp. 672–679).

Efron, M., & Golovchinsky, G. (2011). Estimation methods for ranking recent information. *SIGIR* (pp. 495–504).

Efron, M., Turnbull, D., & Ovalle, C. (2008). University of Texas School of Information at TREC 2007. *Proceedings of TREC 2007*.

Elsas, J. L., Arguello, J., Callan, J., & Carbonell, J. G. (2008). Retrieval and feedback models for blog feed search. *Proceedings of SIGIR 2008* (pp. 347–354).

Ernsting, B., Weerkamp, W., & de Rijke, M. (2007). Language modeling approaches to blog postand feed finding. *Proceedings of TREC 2007*.

Hannah, D., MacDonald, C., Peng, J., He, B., & Ounis, I. (2008). University of Glasgow at TREC 2007: Experiments in Blog and Enterprise Tracks with Terrier. *Proceedings of TREC 2007*.

Hawking, D., & Thomas, P. (2005). Server selection methods in hybrid portal search. *Proceedings of SIGIR 2005* (pp. 75–82).

Inches, G., Carman, M. J., & Crestani, F. (2010). Statistics of online user-generated short documents. *Proceedings of ECIR 2010* (pp. 649–652).

Jones, R., & Diaz, F. (2007). Temporal profiles of queries. *ACM Transactions on Information Systems (TOIS)*, *25*(3), 14.

Kurland, O., & Lee, L. (2005). Pagerank without hyperlinks: structural re-ranking using links induced by language models. *Proceedings of SIGIR 2005* (pp. 306–313). ACM.

Lee, Y., Na, S.-H., Kim, J., Nam, S.-H., Jung, H.-Y., & Lee, J.-H. (2008). Kle at trec 2008 blog track: Blog post and feed retrieval. *Proceedings of TREC 2008*.

Li, X., & Croft, W. B. (2003). Time-based language models. *CIKM* (pp. 469–475).

MacDonald, C., & Ounis, I. (2006a). The TREC Blogs06 collection: Creating and analysing a blog test collection. *Department of Computer Science, University of Glasgow Tech Report TR-2006-224*.

MacDonald, C., & Ounis, I. (2006b). Voting for candidates: adapting data fusion techniques for an expert search task. *Proceedings of CIKM 2006* (pp. 387–396). ACM Press New York, NY, USA.

MacDonald, C., & Ounis, I. (2008). Key blog distillation: ranking aggregates. *Proceedings of CIKM 2008* (pp. 1043–1052).

MacDonald, C., Ounis, I., & Soboroff, I. (2009). Overview of the TREC-2009 Blog Track. *Proceedings of TREC 2009*.

Mei, Q., Zhang, D., & Zhai, C. (2008). A general optimization framework for smoothing language models on graph structures. *Proceedings of SIGIR 2008* (pp. 611–618). ACM.

Mishne, G., & de Rijke, M. (2006). A study of blog search. *Proceedings of ECIR 2006* (pp. 289–301).

Nunes, S., Ribeiro, C., & David, G. (2009). Feup at trec 2008 blog track: Using temporal evidence for ranking and feed distillation. *Proceedings of TREC 2008*.

Ogilvie, P., & Callan, J. (2004). Hierarchical language models for xml component retrieval. *INEX* (pp. 224–237).

Pfahringer, B., Leschi, C., & Reutemann, P. (2007, May). Scaling Up Semi-supervised Learning: An Efficient and Effective LLGC Variant.

Poblete, B., Castillo, C., & Gionis, A. (2008). Dr. searcher and mr. browser: a unified hyperlink-click graph. *Proceedings of CIKM 2008* (pp. 1123–1132).

Robertson, S. E., Maron, M. E., & Cooper, W. S. (1982). The unified probabilistic model for ir. *Proceedings of SIGIR 1982* (pp. 108–117).

Seo, J., & Croft, W. B. (2008). Blog site search using resource selection. *Proceedings of CIKM 2008* (pp. 1053–1062). New York, NY, USA: ACM.

Serdyukov, P., Rode, H., & Hiemstra, D. (2008a). Modeling expert finding as an absorbing random walk. *Proceeding of SIGIR 2008* (pp. 797–798).

Serdyukov, P., Rode, H., & Hiemstra, D. (2008b). Modeling multi-step relevance propagation for expert finding. *Proceeding of CIKM 2008* (pp. 1133–1142).

Soboroff, I., de Vries, A., & Craswell, N. (2007). Overview of the TREC 2006 Enterprise Track.

Sun, A., Hu, M., & Lim, E.-P. (2008). Searching blogs and news: a study on popular queries. *Proceedings of SIGIR 2008* (pp. 729–730). New York, NY, USA: ACM.

Weerkamp, W., Balog, K., & de Rijke, M. (2008). Finding key bloggers, one post at a time. *Proceedings of ECAI 2008* (pp. 318–322).

Zhai, C., & Lafferty, J. (2004). A study of smoothing methods for language models applied to information retrieval. *ACM Transaction of Information Systems*, *22*(2), 179–214.

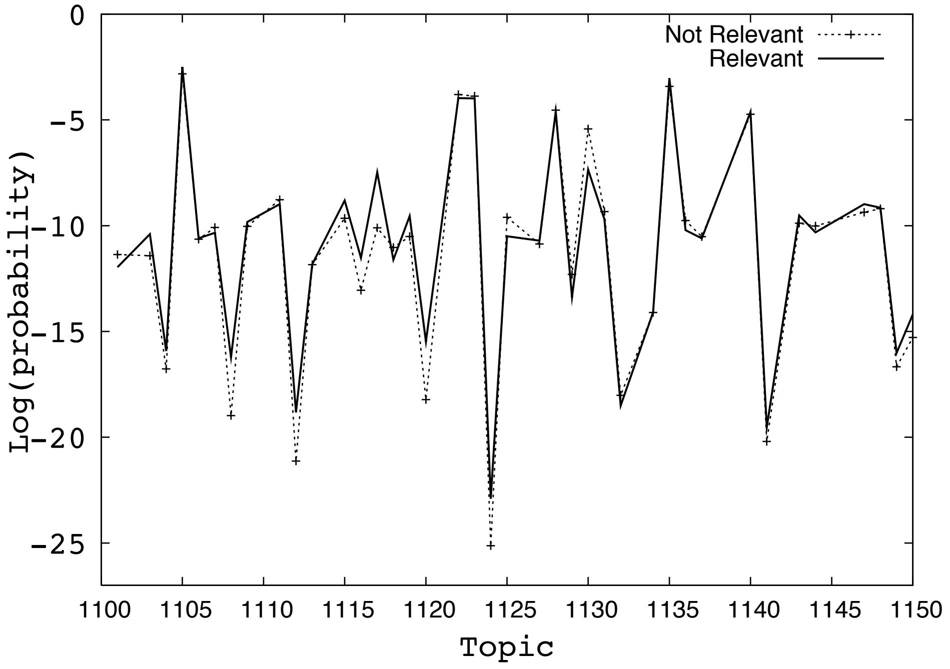

Figure 1: Average score of high ranked posts in relevant and non-relevant blogs for each query

Table 1: Evaluation results for the implemented models over TREC07 query set. Statistically significant improvements over LDM and SDM-Uniform, at the 0.05 level are indicated by † and ‡ respectively. A ⋆ shows statistically significant improvements of a temporal method over its corresponding non-temporal method.

Model	MAP	P@10	Bpref
ExpCombMNZ	0.2649	0.4444	0.3011
ExpCombSum	0.2570	0.4422	0.2880
SDM	0.2580	0.4356	0.2899
LDM	0.2740	0.4489	0.3092
SDM-uniform	0.2905	0.5040	0.3322
RW	0.3029 † ‡	0.5067 †	0.3484 † ‡
Reg	0.3039 † ‡	0.5156 †	0.3448 †
RW-Temporal	0.3081 † ‡	0.5222 † ‡ ⋆	0.3513 †
Reg-Temporal	**0.3125** †	**0.5267** † ‡	**0.3526** † ‡

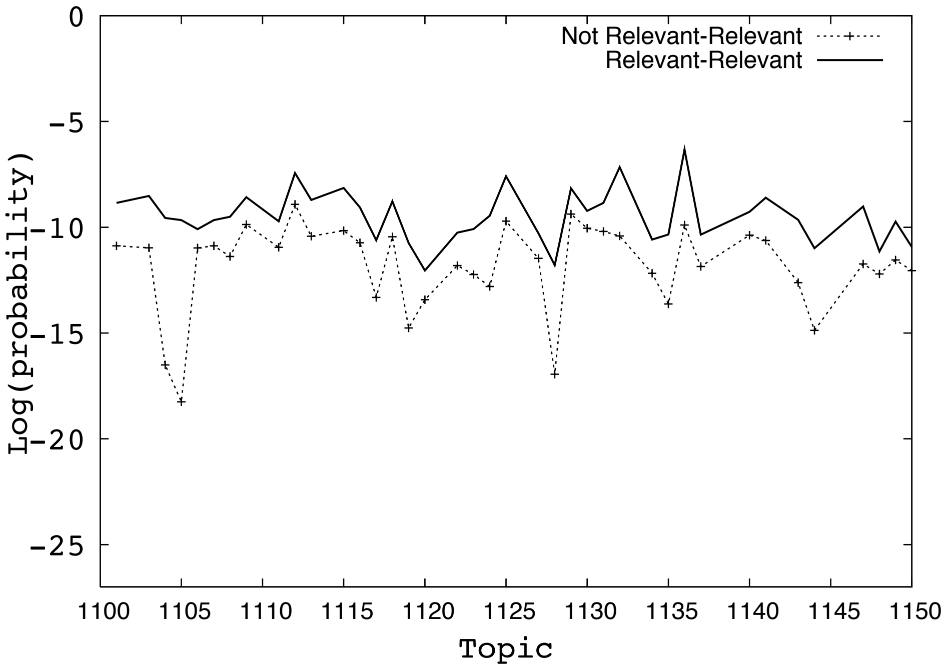

Figure 2: Average similarity between high ranked posts from relevant blogs and high ranked posts from relevant vs. non-relevant blogs

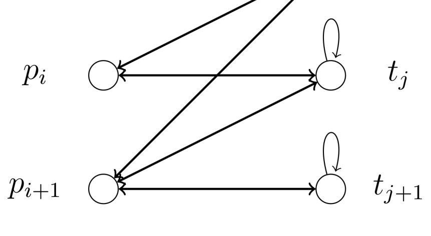

Figure 3: Example of a post-term graph

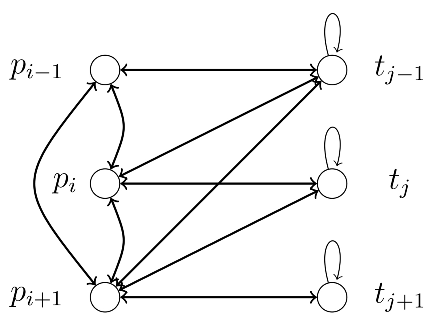

Figure 4: Example of a post-term graph with temporal relations.

Table 2: Evaluation results for the implemented models over TREC08 query set. Statistically significant improvements over LDM and SDM-Uniform, at the 0.05 level are indicated by † and ‡ respectively. A ⋆ shows statistically significant improvements of a temporal method over its corresponding non-temporal method.

Model	MAP	P@10	Bpref
ExpCombMNZ	0.1754	0.2960	0.2205
ExpCombSum	0.1724	0.2980	0.2164
SDM	0.1740	0.3020	0.2171
LDM	0.2096	0.3380	0.2639
SDM-uniform	0.2197	0.3320	0.2705
RW	0.2288 ‡	0.3400	0.2830 ‡
Reg	0.2264 ‡	0.3420	0.2706
RW-Temporal	0.2343 † ‡	0.3480 ⋆	**0.2833** ‡
Reg-Temporal	**0.2377** †	**0.3500**	0.2775

Table 3: Evaluation results for the implemented models over TREC09 query set. Statistically significant improvements over LDM and SDM-Uniform, at the 0.05 level are indicated by † and ‡ respectively. A ⋆ shows statistically significant improvements of a temporal method over its corresponding non-temporal method.

Model	MAP	P@10	Bpref
ExpCombMNZ	0.1814	0.3077	0.2031
ExpCombSum	0.1643	0.2923	0.1942
SDM	0.1682	0.2974	0.1967
LDM	0.1769	0.3026	0.1969
SDM-uniform	0.2046	0.3410	0.2365
RW	0.2131 † ‡	0.3436 †	0.2421 † ‡
Reg	0.2213 †	0.2974	0.2363
RW-Temporal	0.2234 † ‡ ⋆	**0.3462**	**0.2445** † ‡
Reg-Temporal	**0.2268** † ‡	0.3077 ⋆	0.2396

Table 4: Kendall τ correlation coefficient between improvement and the number of relevant blogs for queries

Model	TREC07	TREC08	TREC09
RW-Temporal	0.03	0.03	-0.14
Reg-Temporal	0.11	0.04	-0.08

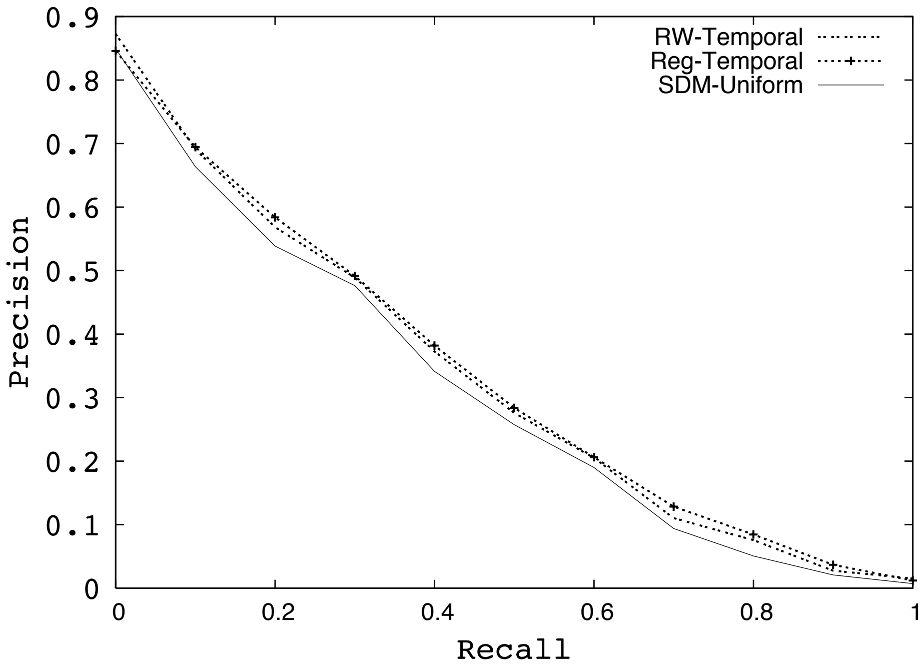

Figure 5: Precision-Recall for TREC07 data set against best performing baseline method

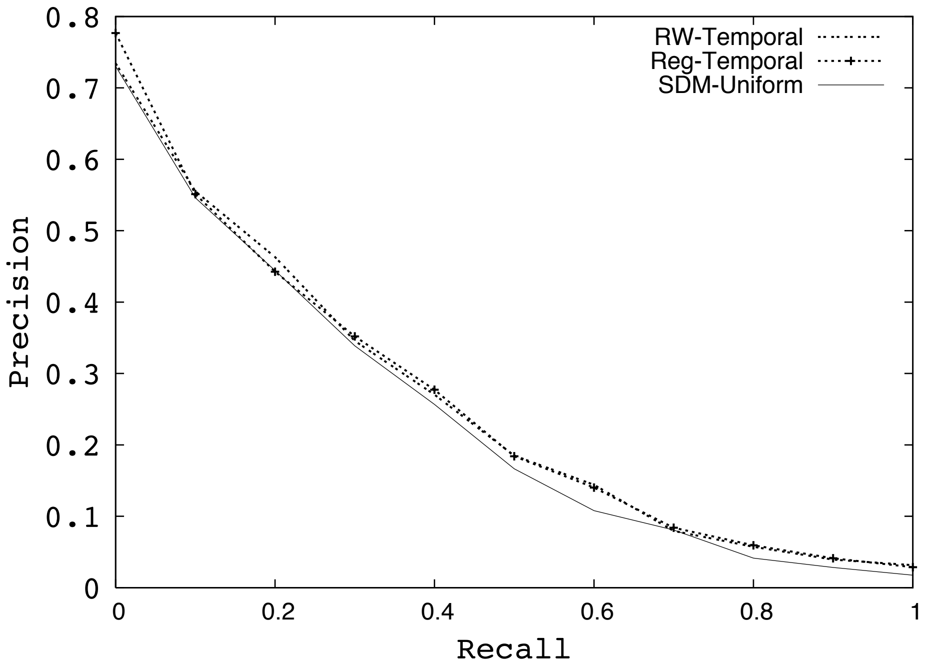

Figure 6: Precision-Recall for TREC08 data set against best performing baseline

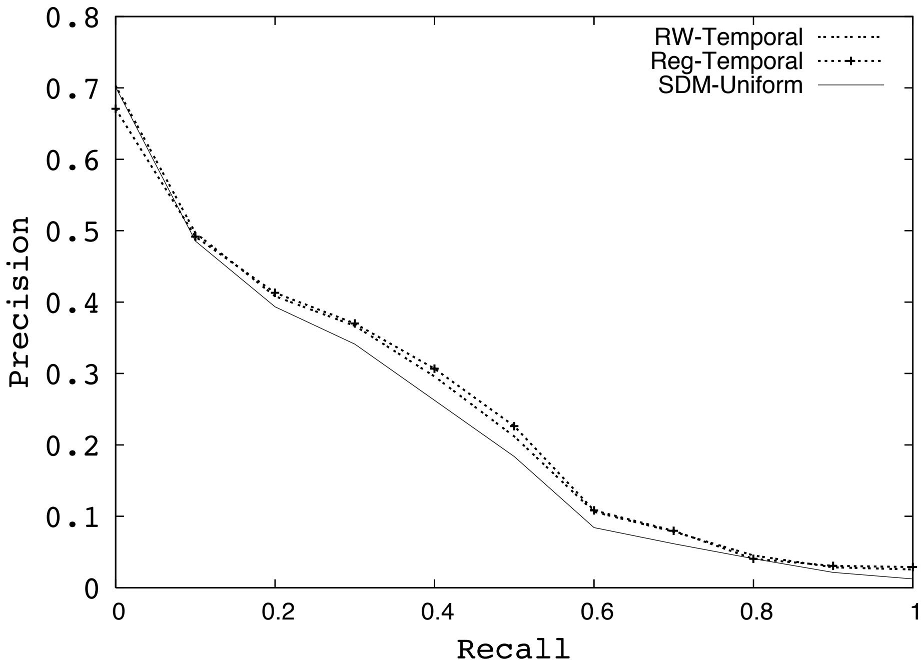

Figure 7: Precision-Recall for TREC09 data set against best performing baseline