Chapter 4

Blog Retrieval: Topical Consistency among Documents

The task on which we focus in this chapter is blog feed retrieval, also called blog distillation [162]. The blog feed retrieval task is defined as identifying blogs that show a central, recurring interest in a given topic. The task has two main characteristics: first, the retrieval units are blogs rather than single posts; second, in order to be considered as relevant, a blog should not just mention the topic of the user query sporadically, but rather it must contain a significant number of posts concerning this topic. An effective approach to blog feed retrieval should take both of these characteristics into account.

Within this context, we investigate whether our coherence score can be exploited to model topical consistency of a blog and thereby improve retrieval effectiveness in the blog feed retrieval task. We start with a brief introduction of the blog feed retrieval task and recall the research questions we have outlined in Chapter 1.

4.1 Introduction

The amount of user generated content available on-line is already voluminous, and it continues to grow on a daily basis. User generated content is not regulated by top-down rules, leaving users free to decide (i) what to write about (topics), (ii) how to write (writing style, language), and (iii) when to write (time of day, regularity). Since user generated content is produced without editorial supervision, standards and conventions that otherwise dictate the form and consistency of written prose, cannot be assumed to be upheld. A specific type of user generated content, blogs (syndicated web journals), has shown a particularly spectacular rise. Currently, bloggers worldwide generate content at a rate in the order of one million new posts per day.\(^1\) With this ever increasing amount of information available in the blogosphere, the need for intelligent access facilities is clear. The information needs of users searching the blogosphere fall into two general categories: the need to find individual blog posts regarding a topic, or the need

\(^1\)http://technorati.com/state-of-the-blogosphere/
to identify blogs that frequently publish posts on a given topic. These categories mirror the short term vs. long term interest distinction observed by Mishne and de Rijke [180] in their study of blog search behavior. Although currently most focus is on finding blog posts, some systems offer the possibility to search for full blogs, alongside post-level retrieval functionality [72]. Searchers can use blog search to identify blog feeds they would like to add to their feed readers.

Two key features set blog content apart from conventional web content and necessitate that dedicated retrieval algorithms and approaches be developed for blogs. The first is the strong social aspect of blog content, most readily noticeable in the use of blog rolls, user assigned tags and, especially, comments to posts. The second, and the one most relevant to the current context, is the noisiness of the data in the blogosphere. We identify two levels at which blog content is noisy: (i) the blog post level and (ii) the blog level. At the post level, noise expresses itself in unexpected language usage, spelling and grammatical errors, non-language characters (e.g., emoticons), and mixed data types (pictures, video, text). At the blog level, the noise can be characterized as topological noise, which tends to be semantic rather than lexical or structural. A blog can (and most likely will) be about different topics. As an illustration of different levels of topological noise in blogs, consider Figures 4.1(a) and 4.1(b), where two blogs treating the subject of vegetable gardening are displayed in the NetVibes feedreader. In the blog in Figure 4.1(b), the blogger digresses from the topic of vegetable gardening to write about other topics. Dealing effectively with this type of topological noise is critical for improving performance on blog feed retrieval, since blogs with topological noise show less consistent interest in particular subjects and are therefore a priori less likely to be appreciated by users in the setting of the blog feed retrieval task.

How can we measure topological noise? Specifically, how can we measure it in blogs? The characteristics of the blog feed retrieval task combined with the challenge presented by noisy data require an approach that is both flexible and sufficiently robust. We view blog feed retrieval as an association finding task: which blogger is most closely associated with the given topic? And: how consistent is this blogger regarding the topic? To address the first issue, we adopt the language modeling approach used in expert retrieval [13, 256]. To tackle the second issue—the core issue addressed in this chapter—, we integrate the coherence score into this language modeling-based approach. The coherence score measures the topological clustering structure of a blog. Loose clustering reflects topological diffuseness and signals the presence of topological noise in the blog as defined in Chapter 3. In contrast, tight clustering indicates that the blog remains focused on one or a few central themes.

Given these issues, we explore the following three dimensions in this chapter, which we formulate as research questions:

**RQ2a.** How do we measure topological consistency for a blog?

**RQ2b.** How can we use the coherence score in our blog retrieval process?

---

[^2]: http://www.netvibes.com
4.1. Introduction

**RQ2c** Given that the collection we use in our experiments only provides us with a sample of blog posts generated by the underlying blog models, how does the sample size influence the estimation of the coherence and how does this influence blog feed retrieval?

For RQ2a, we offer our coherence score as a solution; we compare it against lexical cohesion, a standard measure for determining the diversity of topics discussed in a text. For RQ2b, we compare a number of options, ranging from treating the coherence score as a simple prior to modeling it as a multiplicative factor whose contribution is a function of the retrieval status value of a blog. The final question RQ2c is addressed using an experimental exploration.

Our main finding is that our proposed coherence score can estimate the topical noise present in blogs. Moreover, it can help combat the topical noise present in blogs when it is weighted with the initial retrieval score, preventing blogs that display tight topic structure, but that are not relevant to the query, from rising to the top of the result list. In addition, we find that a minimum of 20 posts is required to get a proper estimate of the coherence of a blog, regardless of the actual size of the blog. This finding is supported by blog feed retrieval results: The coherence score reaches its optimal performance increase when a substantial number of posts (> 20) have been written in a blog.

The remainder of the chapter is organized as follows: In Section 4.2 we study our proposed coherence score as compared against the well-known lexical cohesion measure. In Section 4.3 we detail the modeling of blog feed retrieval and the integration.
of coherence in this framework. Section 4.4 specifies our experimental settings, and we discuss the results of the experiments in Section 4.5. In Section 4.6 we analyze our experimental findings, before concluding in Section 4.7.

4.2 Topical consistency measures

This section discusses two methods of capturing the topical consistency of a text. In Chapter 3 we have already introduced our coherence score as a measure of topical coherence. Here, we introduce lexical cohesion, a familiar text analysis approach that uses information about the semantic relatedness of words to capture the topical structure of a text. We then compare our coherence score against lexical cohesion. Evidence emerges that the advantages of lexical cohesion are outweighed by its shortcomings. In particular we comment on its lack of sensitivity to topical hierarchy.

4.2.1 Lexical cohesion

The concept of cohesion [82] is used in text analysis to describe the topical relationships between various units of text. Cohesion is a set of characteristics that conspire to make text “stick together” topically [15]. Lexical cohesion measures cohesion by examining the semantic relationships between the content words used in a text [182]. Lexical cohesion is easy to identify [15] and can be calculated automatically using an appropriate linguistic resource such as a thesaurus. Semantically similar words (usually nouns) occurring in close proximity to one another build lexical chains, which indicate that a unit of text is about the same topic [182]. Lexical chains form the basis for models of lexical cohesion [15, 182, 236]. A primitive form of lexical cohesion does not make use of similar lexical words, but rather measures repetition of the same word form or forms. The cohesiveness filter proposed by Amitay et al. [8] encodes an entropy-based measure of query-word repetition patterns and, is an example of this primitive form of lexical cohesion. The disappointing results of this filter as applied to the task of identifying topically focused web pages in the TREC-2003 Web Track topic distillation task motivate us to turn our consideration to full-fledged forms of lexical cohesion that look beyond word-form repetition and make use of external resources to derive information concerning lexical similarity.

A priori, lexical cohesion is an appealing approach to capturing topical consistency. It is intuitive that the topical diversity of a text is reflected in the number of distinct topics it discusses. The number of topics in a text, in turn, is reflected by the number of lexical chains of words with similar meanings that the text contains. The low-noise blog excerpt in Figure 4.1(a) and the moderate-noise blog excerpt in Figure 4.1(b) are convenient examples that provide an impression of how lexical chains capture topical consistency. We generate strong lexical chains from these blog excerpts using the LexicalChain application of the Electronic Lexical Knowledge Base (ELKB).\(^3\) which

\[\text{http://www.nzdl.org/ELKB}\]
4.2. Topical consistency measures

is based on Roget’s Thesaurus and implements the algorithm proposed by Barzilay and Elhadad [15]. The chains are shown in Table 4.1. The LexicalChain algorithm computes lexical chains by clustering words that are both semantically similar and near to each other in the text. *Chain score* is the length of the chain as measured by the number of words it contains weighted with a factor reflecting the number of repeated words. *Strong chains* are defined as chains that have a score greater than the mean score plus two standard deviations. The highest frequency member of a chain is defined to be its *keyword*. From Table 4.1, it can be observed that the low-noise blog excerpt generates eight strong lexical chains, seven of which have a unique keyword. The moderate-noise blog, on the other hand, generates nine strong lexical chains. The difference in chain number reflects human intuitions about the topical diversity of the two blogs. The difference is not strikingly large, but still serves to illustrate the way in which intuitions of topical diversity are related to the number of topics as reflected by the number of lexical chains a text contains. Other five post excerpts of the same blogs display similar differences in chain number.

In addition to providing an impression of how lexical cohesion works, this example also illustrates one of its shortcomings. Lexical cohesion is sensitive to the progression of topics in a text, but is rather blind to their hierarchical structure. Where humans may differentiate between a central and a subordinate topic, the LexicalChain algorithm produces two lexical chains of approximately the same length. For example, in Table 4.1 it can be seen that in the low-noise topical blog, a chain with the keyword “soil” is produced, which is a plausible central topic of the blog. A chain with the keyword “space” is also produced, which arises due to mention of spatial concepts in various contexts, but is less likely to be understood as an actual topic of the blog. It is challenging to determine the topical consistency of a text collection by using lexical chains to count the number of distinct topics occurring, since it is not readily obvious which chains to count as representing central topics of the text.

The problem of distinguishing central from subordinate topics can be circumvented by setting aside the chain-based lexical cohesion approach, and instead looking directly at the inherent clustering structure of the collection, i.e., the topic groups that emerge when the documents in the collection are compared to each other. That said, the coherence score we proposed in Chapter 3 is such a measure that captures the clustering structure of the collection. In the next section, we revisit the coherence score and compare it to lexical cohesion.

4.2.2 Coherence score versus lexical cohesion

In Section 3.3 on page 40 we have seen that the coherence score is able to capture the clustering structure of data, and in particular, the topical consistency of text. The coherence score holds clear potential for capturing the topical consistency of user generated content. Here, in order to get a direct impression of the effectiveness of the coherence score as compared to the lexical cohesion, we calculate the coherence score for the blog examples discussed above. The two blogs generate coherence scores consistent
Chapter 4. Blog Retrieval: Topical Consistency among Documents

Table 4.1: The low-noise blog excerpt (cf. Figure. 4.1(a)) generates seven unique strong lexical chains, while the moderate-noise excerpt (cf. Figure. 4.1(b)) generates nine. Chains are ordered by decreasing chain strength; keywords are shown in bold.

with our expectations. The coherence score of the blog excerpt with low topical noise (cf. Figure. 4.1(a)) is 0.5 compared to 0.3 for the blog excerpt with moderate topical noise (cf. Figure. 4.1(b)).

What are the advantages of using coherence score as a measure of topical consistency for blogs compared to lexical cohesion? First, the coherence score relies only on the statistics derived from the collection and is independent of any external resources. In order to calculate alternate measures such as lexical cohesion, an external knowledge resource such as a thesaurus or lexical database such as WordNet [66] is necessary. Dependence on external resources raises several issues. External resources often fail to be up-to-date with regard to proper nouns [236], which is especially needed in a fast-changing environment like the blogosphere. Further, we must be able to filter our collection and regard blogs written only in languages covered by available resources, a challenging task in face of the fact that some bloggers switch languages while posting. In these respects, using the coherence score offers clear benefits of independence and flexibility.

Second, the coherence score does not require optimization of parameter settings. For the coherence score, the only parameter is the threshold $\tau$, which defines the “non-randomness” for a given collection. Recall that $\tau$ is set by sampling the background collection for a given $\kappa$. Although $\kappa$ is determined heuristically, our previous experiments show that the value 0.05 is quite stable. Coherence is thus easier to apply than measures such as lexical cohesion. In order to build lexical chains, the setting of two parameters is required: a threshold on the semantic relatedness of two words and a threshold on the physical distance, i.e., the number of words separating them in the running text. These parameters determine whether a word should be added to an existing chain or start a new chain [236]. Presumably, parameter settings would have to be
4.3. Using coherence in the setting of blog feed retrieval

<table>
<thead>
<tr>
<th>Total number of blogs:</th>
<th>83,320</th>
</tr>
</thead>
<tbody>
<tr>
<td>blog length</td>
<td>&lt; 10</td>
</tr>
<tr>
<td>number of blogs</td>
<td>21,290</td>
</tr>
</tbody>
</table>

Table 4.2: Distribution of the blog lengths, i.e., number of posts contained in a blog.

re-optimized for a new corpus.

Third, the coherence score directly captures the clustering structure of the collection. For this reason, it is not necessary to be concerned about identifying individual topics or their relative importance in the blog. As discussed above, a lexical cohesion measure based on lexical chains encounters the challenge of distinguishing chains representing central topics from chains representing subordinate topics. Although we do not exclude the possibility that further development work would allow this issue to be addressed, the coherence score approach offers the advantage of circumventing the issue entirely.

Fourth, the coherence is more efficient to compute. Its computational complexity is \(O(s \cdot n^2)\), while the complexity of a typical lexical chain algorithm is \(O(s^2 \cdot n^2)\), where \(s\) is the average length of the individual documents in words and \(n\) is the number of documents in the document set on which the coherence measure is performed. Although in practice the computational complexity of the calculation of lexical chains can be kept well below its theoretical limit, it still fails to be competitive with that of the coherence score. For our experiments, we calculate the coherence score for the set of blog posts in a given blog. The coherence score is calculated offline at indexing time, i.e., we calculate the scores once for all blogs in the collection. With our implementation, the calculation of pairwise cosine similarity scores takes around 1.5 seconds for 500 documents. Table 4.2 shows the distribution of the number of posts in blogs, which provides an impression of the feasibility of our approach.

Finally, given the definition of the coherence score in Chapter 3, the application of coherence score is not even limited to text data. Information other than words such as the structure of the documents, hyperlinks contained in the webpages, etc., could be easily integrated.

These advantages provide motivation for us to leave aside consideration of measures with the disadvantages of lexical cohesion and continue our investigation by testing the efficacy of the coherence score. In particular, we investigate whether the coherence score can be exploited to model topic consistency and improve retrieval in the blog feed retrieval task.

4.3 Using coherence in the setting of blog feed retrieval

In this section we detail the modeling of the task we address: modeling topical noise in user generated content. To this end, we first explain our blog feed retrieval modeling framework in Section 4.3.1; after that we introduce alternative ways of incorporating
the coherence score in this framework (Section 4.3.2).

4.3.1 Blog retrieval model

Our approach to modeling blog feed retrieval, first introduced in [13], is based on expert retrieval models [12]. As indexing unit we use individual blog posts. We have three reasons for this: (i) to allow for easy incremental indexing, (ii) for presentation of retrieval results posts are natural units, and (iii) the most important reason, to allow the use of one index for both blog post and blog feed search [256].

We adopt a probabilistic approach to the task of determining relevance of blogs to the user query and formulate the task as follows: what is the probability of a blog being relevant given the query topic $q$? In other words, we estimate $p(\text{blog}|q)$, and rank blogs according to this probability. Since a query generally consists of only a few terms, often under-representing the information need that gave rise to it, Bayes’ Theorem is applied in order to achieve a more accurate estimate:

$$p(\text{blog}|q) = \frac{p(q|\text{blog}) \cdot p(\text{blog})}{p(q)},$$

(4.1)

where $p(\text{blog})$ is the probability of a blog; in our baseline approach $p(\text{blog})$ is assumed to be uniform, that is $p(\text{blog}) = |\text{blog}|^{-1}$, where $|\text{blog}|$ is the number of blogs in the collection; other ways of estimating $p(\text{blog})$ are detailed in Section 4.3.2. The component $p(q)$ indicates the probability of a query. In the remainder of the chapter, we refer to the Retrieval Status Value (RSV) rather than to $p(\text{blog}|q)$. This terminological shift is necessary since our experiments involve incorporating scores into $p(\text{blog}|q)$ that have the same scale as probabilities, but are not otherwise true probabilities.

Following a common practice in language modeling approaches, $p(q)$ is discarded as it does not affect the ranking of the results (for a given query $q$). However, when the impact of the coherence score is taken to be a function of the RSV (as we will discuss in Section 4.3.2), the normalization term is necessary in order to ensure that the weight of the coherence score is compatible across queries. A non-normalized $\text{RSV}$ will impose an unwanted limitation of the domain and thereby also the range of the coherence score function.

In our experiments, we apply the full Bayes’ Theorem, which leads to the estimation of the probability $p(q)$. To estimate $p(q)$ we adopt the method used by Lavrenko and Croft [147], who estimate the probability of a term $p(w)$ is with the following equation:

$$p(w) = \sum_{m \in M} p(w|m)p(m),$$

(4.2)

where $w$ is a term and $M$ is a set of language models derived from top ranked documents. We can translate this equation to our blog feed retrieval model by replacing $p(w)$ with $p(q)$ and $M$ with $B$, a set of blogs. We end up with Eq. 4.3:

$$p(q) = \sum_{\text{blog} \in B} p(q|\text{blog})p(\text{blog}).$$

(4.3)
4.3. Using coherence in the setting of blog feed retrieval

We set $B$ to be the top 200 results, i.e., retrieved blogs, for query $q$ so as to estimate $p(q)$. Next, we focus on the estimation of the query likelihood, $p(q|\text{blog})$: the likelihood of the topic expressed by the query $q$ given a blog. Query likelihood estimation is accomplished using standard language modeling techniques. We build a textual representation of a blog based on posts that belong to the blog. From this representation we estimate the probability of the query topic given the blog’s model. The language modeling framework makes it possible to use blog posts to build associations between queries and blogs in a transparent and principled manner.

Our model represents a blog using a multinomial probability distribution over a vocabulary of terms. For each blog, a blog model $\theta_{\text{blog}}$ is inferred, such that the probability of a term $t$ given the blog model is $p(t|\theta_{\text{blog}})$. The model is then used to predict the likelihood that a blog gives rise to a particular query $q$. We make the assumption that each query term can be assumed to be sampled identically and independently from the blog model. Applying this assumption, the query likelihood is obtained by multiplying the likelihoods of the individual terms contained in the query:

$$p(q|\theta_{\text{blog}}) = \prod_{t \in q} p(t|\theta_{\text{blog}})^{n(t,q)}, \quad (4.4)$$

where $n(t,q)$ is the number of times term $t$ is present in query $q$. In order to prevent data sparseness from resulting in zero query likelihoods, we follow standard procedure and smooth the query likelihood model. The maximum likelihood estimate of the probability of a term given a blog $p(t|\text{blog})$, which is then smoothed with term probabilities $p(t)$ estimated using the background collection:

$$p(t|\theta_{\text{blog}}) = \lambda_{\text{blog}} \cdot p(t|\text{blog}) + (1 - \lambda_{\text{blog}}) \cdot p(t). \quad (4.5)$$

In Eq. 4.5, $p(t)$ is the probability of a term in the document repository. The effect of smoothing is to add probability mass to the blog model in proportion to how likely that blog is to be generated (i.e., published) by a generic blogger. We discuss the estimation of the smoothing parameter $\lambda_{\text{blog}}$ in Section 4.4.

The individual blog posts act as a bridge to connect $t$ and the blog, resulting in the following estimate of $p(t|\text{blog})$:

$$p(t|\text{blog}) = \sum_{\text{post} \in \text{blog}} p(t|\text{post},\text{blog}) \cdot p(\text{post}|\text{blog}). \quad (4.6)$$

We make the assumption that the post and the blog are conditionally independent, setting $p(t|\text{post},\text{blog}) = p(t|\text{post})$. The importance of a given post within the blog is expressed by $p(\text{post}|\text{blog})$. A simple approach to estimating this value is to assume a uniform distribution, i.e., all posts of a blog are weighted equally in terms of importance. Under this assumption, $p(\text{post}|\text{blog}) = \text{posts}(\text{blog})^{-1}$, where $\text{posts}(\text{blog})$ is the number of posts in the blog.
4.3.2 Incorporating the coherence score into blog retrieval

Now that we have outlined our blog retrieval framework, we shift our attention to the incorporation of the coherence score in this framework. Before we jump to actually modeling this, we take a step back and look at the relation between the coherence of a blog and its relevance regarding a topic. In case of a (topically) relevant blog, this blog should not be highly favored in the final ranking unless it is also topically coherent. On the other hand, if we have a blog that has high topical coherence because it consistently treats a different topic than the relevant topic, we do not want this blog to enjoy an unjustified promotion within the final ranking. Instead, we would like to target a more desirable behavior: blogs that are ranked high for a given topic should enjoy a boost from the coherent score that allows them to maintain their prominence while bottom ranked blogs should be prevented from deriving benefit from their coherence score; in the latter case the chance is greater that they are coherent with respect to non-relevant topics. Finally, documents in between should be given a moderate advantage if their coherence scores are high. We can look at this desirable behavior as local re-ranking in contrast to global re-ranking, which allows for a document to take a brutal jump from the very bottom to the very top of the final ranking.

A transparent, straightforward integration of coherence in our retrieval framework can be implemented by taking the coherence score of a blog to supply information about query-independent blog relevance, encoded by the blog prior \( p(\text{blog}) \). As detailed in Section 3.1 on page 36, the coherence score is already a proportion, which means that it is scaled like a probability, and for this reason we can simply estimate

\[
p(\text{blog}) = \text{Co}(\text{blog})
\]

where \( \text{Co}(\text{blog}) \) is calculated using Eq. 3.2 on page 36, and the threshold \( \tau \) is estimated to be 0.1, given that \( \kappa \) is set to 0.05 heuristically. In cases where the coherence score of a blog is zero, or when no coherence can be calculated (in the case of one-post blogs), we assign a low probability (0.01). On the one hand we do not want zero probabilities, but on the other we believe these blogs should not receive a high prior probability, since they do not show the recurring interest in a topic.

Although the implementation of coherence as a prior is straightforward, it does not fulfill the properties we discussed in the first paragraph of this section: topically more relevant blogs should receive a solid boost if coherent, less relevant documents should not be affected. In fact, this boils down to weighting the coherence score by some notion of topical relevance. One issue here is that we do not have relevance judgements for our ranked documents. Instead, we use the baseline retrieval score \( RSV \) of a blog with a uniform prior (viz. Eq. 4.1), as a substitute for judged relevance. We prefer the retrieval score of the blog over an obvious alternative, using the rank of the blog in the retrieval result list. If the rank were used, a small difference in \( RSV \) could have a disproportionately large impact on the rank, making the weights over-sensitive and unreliable.

In order to capture the desideratum that more relevant blogs receive a bigger boost
from the coherence score, the weights are functions of $RSV$, the baseline retrieval score, and are designed to be monotonically increasing. In particular, we want blogs with $RSV$'s close to 0 to receive nearly no contribution from the coherence score while blogs with the highest $RSV$'s should receive the full impact from the coherence score. The following functions modify the relation between the coherence weight ($W(\cdot)$) and the $RSV$ in a manner consistent with these requirements. We have selected these functions to represent the range of possible relations between $RSV$ and coherence score that we believe could potentially be useful.

**Linear function (lin)**

$$W(RSV) = RSV$$  \hspace{1cm} (4.8)

**Normal distribution (norm)** with $\mu = 1$ and $\sigma$ as a free parameter:

$$W(RSV) = \frac{1}{\sigma \sqrt{2\pi}} \exp \left( -\frac{(RSV - \mu)^2}{2\sigma^2} \right)$$  \hspace{1cm} (4.9)

**Quadratic function 1 (quad1):**

$$W(RSV) = RSV^2$$  \hspace{1cm} (4.10)

**Quadratic function 2 (quad2):**

$$W(RSV) = -(RSV - 1)^2 + 1$$  \hspace{1cm} (4.11)

**Mixed function of 4.10 and 4.11 (qmix)** with $\alpha$ as the free parameter:

$$W(RSV) = \begin{cases} 
RSV^2 & \text{if } RSV < \gamma, \\
-(RSV - 1)^2 + 1 & \text{otherwise.}
\end{cases}$$  \hspace{1cm} (4.12)

This choice of functions allows us to explore a linear relation (Eq. 4.8), a non-linear relation with different rates of increase (Eq. 4.9, 4.10, 4.11), and a combination of different rates of increase (Eq. 4.12). Figure 4.2 shows the curves of these functions in order to provide an intuition of the properties of the functions.

Finally, the weighted coherence score of a blog for a given query is defined as:

$$wCo(blog, query) = W(RSV) \cdot Co(blog)$$  \hspace{1cm} (4.13)

The experimental models use $wCo$ as the blog “prior.” Substituting it for $p(blog)$ in Eq. 4.1 leaves us with the final ranking equation

$$RSV = \frac{p(q|blog) \cdot wCo(blog, q)}{p(q)}.$$  \hspace{1cm} (4.14)

In summary, from our observations on the relation between coherence and relevance, we introduce two main methods for incorporating the coherence score into our retrieval framework: (i) a query-independent method, using $Co(blog)$ directly as $p(blog)$, and (ii) a relevance-dependent method, where $Co(blog)$ is weighted using a function of the $RSV$. The latter method is translated into five weighting functions.
Figure 4.2: Weighting functions.

### 4.4 Experimental setup

Our next aim is to compare the effectiveness of the blog retrieval methods just described. In particular, we aim to answer the following research questions as introduced in Section 4.1:
4.4. Experimental setup

RQ2a. How do we measure topical consistency for a blog?

RQ2b. How can we use the coherence score in our blog retrieval process?

RQ2c. Given that the collection we use in our experiments only provides us with a sample of blog posts generated by the underlying blog models, how does the sample size influence the estimation of the coherence and how does this influence blog feed retrieval?

Before answering these research questions, we detail our experimental setup.

4.4.1 Collection

For our experiments on blog feed retrieval we use the TRECBlog06 collection [161]. The TRECBlog06 corpus was collected by monitoring feeds (blogs) for a period of 11 weeks and downloading html documents behind all permalinks. For each permalink (or blog post or document) the blog ID is registered. For our experiments we did not make use of the syndication information (i.e. RSS feeds). The collection contains 3.2 million blog posts gathered from 100K blogs.

The TREC 2007 Blog track supplies 45 blog feed retrieval topics, also referred to here as queries, and assessments concerning which blogs are relevant to which topics [162]. Topic development and assessment annotation were carried out by the participants of the track. In order to determine the relevance of a blog to a topic, assessors were asked to confirm that a substantial number of blog posts did indeed deal with that topic. For all our runs we make use of the topic field (T) of the topics and discard the longer formulations of the topics (i.e., those contained in the description (D) and narrative (N) fields).

4.4.2 Evaluation metrics and significance

In order to measure the performance of our approach to modeling topical noise in blog distillation, we use mean average precision (MAP) as well as three precision-oriented measures: precision at ranks 5 and 10 (P@5, P@10), and mean reciprocal rank (MRR).

We determine statistical significance of differences using a two-tailed paired t-test with $\alpha = .05$. Significant changes are indicated using $\uparrow$ (significant increase) or $\downarrow$ (significant decrease).

4.4.3 Smoothing

The performance of language modeling-based retrieval methods is highly responsive to smoothing [268]. To estimate the smoothing parameter $\lambda_{\text{blog}}$ in Eq. 4.5 in our model, we set $\lambda_{\text{blog}}$ equal to $\frac{n(\text{blog})}{\beta + n(\text{blog})}$, where $n(\text{blog})$ is the length of the blog (i.e., we sum the lengths of all posts of the blog). Essentially, the amount of smoothing applied to a given
Chapter 4. Blog Retrieval: Topical Consistency among Documents

blog model is proportional to the length of that blog (and is like Bayes smoothing with a Dirichlet prior [163]). This approach is consistent with the observation that if a blog contains only few posts, estimation of the blog model is less robust and background probabilities are relatively more reliable and should thus make a larger contribution to the model. We set \( \beta \) to be the average blog length in the test collection (here, \( \beta = 17,400 \)).

4.4.4 Parameter estimation

For the functions \textit{norm} and \textit{qmix} we need to set parameters \( \sigma \) and \( \gamma \). We performed a sweep over possible (and sensible) values of both parameters (\( 0 < \sigma < 1.0; 0 < \gamma < 0.1 \)) and evaluated the performance on MAP. Based on the results of the sweep, we select \( \sigma = .05 \) for \textit{norm} and \( \gamma = .05 \) for \textit{qmix}. Note that we are not trying to optimize the performance by selecting the best parameter, rather, we want to see the impact of the model parameter on the retrieval performance. For this reason, the generalization ability of the parameter setting is not considered.

4.5 Results

Let us revisit our research questions. For our first question, \textit{How do we measure topical consistency for a blog?}, we offer our coherence score as a solution.

In this section, we turn to the second question, \textit{How can we use the coherence score in our blog retrieval process?} A number of options, ranging from treating the coherence as a simple prior to modeling it as a multiplicative factor whose contribution is a function of the \textit{RSV} of a blog, are proposed in Section 4.3. We now compare the results of these options, analyze the outcomes. In Section 4.6, we look into the third research question of \textit{how the sample size influences estimating the coherence score of a blog and what the impact is on blog feed retrieval}.

Table 4.3 lists the results for our baseline model, \textit{baseline}, which uses a uniform prior, our straightforward implementation of coherence, \textit{prior}, which uses \textit{Co(blog)} (cf. Eq. 3.2) as prior, and the five experimental models, designated \textit{lin}, \textit{norm}, \textit{quad1}, \textit{quad2}, and \textit{qmix} according to which version of the weighted coherence score \textit{wCo} they integrate.

The run using coherence as a prior performs significantly worse than the baseline in terms of MAP, but shows slight (non-significant) improvements on early precision (P@5) and MRR. We can see that all weighting functions show some improvement over the baseline, with \textit{qmix} performing best in terms of MAP and MRR. The improvement gained over the baseline by applying this function as a weight to the coherence score is significant. We can see that the coherence score does not only help MAP and MRR, but also shows improvements in terms of P@5 and P@10 in most cases, although not significant.
4.5. Results

<table>
<thead>
<tr>
<th>Model</th>
<th>MAP</th>
<th>P@5</th>
<th>P@10</th>
<th>MRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>.3272</td>
<td>.4844</td>
<td>.4844</td>
<td>.6892</td>
</tr>
<tr>
<td>prior</td>
<td>.2945 ▼</td>
<td>.5022</td>
<td>.4822</td>
<td>.6959</td>
</tr>
<tr>
<td>lin</td>
<td>.3326</td>
<td>.5022</td>
<td>.5067</td>
<td>.7266</td>
</tr>
<tr>
<td>norm</td>
<td>.3325 ▲</td>
<td>.5022</td>
<td>.4822</td>
<td>.7103</td>
</tr>
<tr>
<td>quad1</td>
<td>.3327</td>
<td>.5022</td>
<td>.5067</td>
<td>.7377</td>
</tr>
<tr>
<td>quad2</td>
<td>.3365 ▲</td>
<td>.5022</td>
<td>.5022</td>
<td>.7154</td>
</tr>
<tr>
<td>qmix</td>
<td>.3382 ▲</td>
<td>.5067</td>
<td>.5022</td>
<td>.7394 ▲</td>
</tr>
</tbody>
</table>

Table 4.3: Results of coherence score, implemented as prior, and using linear function (lin), normal distribution (norm), quadratic function 1 (quad1), quadratic function 2 (quad2), and the combination of quadratic function 1 and 2 (qmix). Significance computed against the baseline.

Figure 4.3: Impact of model parameters on retrieval performance.

As described in Section 4.4.4, we set the parameter values of norm and qmix, namely $\sigma$ and $\gamma$, by sweeping over a range of possible values. In order to see the impact of these parameters on the end-to-end retrieval performance, in Figure 4.3 we illustrate the relation between the values of the model parameters and the retrieval performance in terms of MAP. We see that for norm, with the change of $\sigma$, MAP reaches a global maximum at $\sigma = 0.05$, and afterwards, the MAP scores decrease slightly without dramatic changes. For qmix, the MAP scores across different values of $\gamma$ do not differ significantly in general and a global peak at $\gamma = 0.05$ can be found. Note that the y-axis of Figure 4.3(a) and that of Figure 4.3(b) have different ranges. In addition, since the change of MAP scores remains below 0.001 for $\gamma \in [0.1, 0.9]$, in Figure 4.3(b) we only show the MAP scores for $\gamma \in [0.001, 0.1]$ where the change is relatively more obvious.

Let us take a closer look at the results per test query. In Figure 4.4 we compare the performance of each of the models to the baseline and plot the increase or decrease in AP for each query. The plots show that (i) norm increases performance in 31 of
45 queries, but gains are moderate, (ii) *quad1* hurts more queries than it improves (23 vs. 22), (iii) the same goes for *lin* (again 23 vs. 22), (iv) in both cases the maximum increase in AP is high (.15 for query 974), but so is the maximum drop (-.14 for query 979), (v) *quad2* improves performance in 34 of 45 queries, but also shows a large drop for several queries, and finally (vi) *qmix* improves over the baseline in 35 of 45 queries, with a limited drop in AP for the worst performing query (-.07 for query 979). The query that improves most after integrating the coherence score into the model is query 974 (*tennis*), for all models. Query 979 has the worst performance (*lighting*), for all models. Queries whose performance neither improved nor degraded include query 951 (*mutual funds*), query 969 (*planet*), and query 933 (*buffy vampire slayer*). We hypothesize that the potential of the coherence score to improve retrieval performance for a query is (i) related to the breadth of the vocabulary that a blogger uses to discuss the query, (ii) the ability of the query to inspire bloggers over time and (iii) spam blogs whose word distributions cause them to be relevant to that query.

What happens when we explore the per-query differences between the run using coherence as a prior and the runs using the weighting functions? Three queries score worse using weighting functions compared to the prior: 953 (*biofuels may damage forests*), 957 (*Russia*), and 992 (*copyright law*). On the other hand we see three queries that are in the top 3 of most improved queries over the prior run (for all weighting functions): 974 (*tennis*), 973 (*autism*), and 954 (*Mac*). In general, very few queries actually perform better in the prior run than using the weighting functions (8-12 queries out of 45).

Finally, we look at the differences between the runs using the various ways of weighting coherence and see what causes the final evaluation results to be different: Are certain queries hurt by one function, but improved by another? Or do we see a general trend of queries improving or dropping for all functions, just differing in the degree of gain or loss? We try to answer this question using several queries as examples. First, queries 979 and 982 drop most and second most for all functions. At the other end of the spectrum, we have a similar, consistent behavior for queries 974 (improves most), 994 (improves second most), and 995 (improves third most). Only few queries show different behavior: query 964 (*violence in sudan*) improves for *norm*, *qmix*, and *quad2*, but drops for *lin* and *quad1*. Also, query 992 (*copyright law*) drops in all cases, except for *norm*. The overall picture however, shows consistent behavior for queries over all functions, with the level of improvement (or loss) making up for the differences in MAP between the runs.

### 4.6 Further discussion

Following the assumption we made in our blog retrieval model, for each blog there is a blog model that generates the texts we observe. Since the Blog06 collection is crawled in a certain period, for a given blog we can see it as a sample drawn from an underlying distribution generated according to the blog model. One would expect that the blogs
Figure 4.4: AP differences between baseline and (left-to-right, top-to-bottom) coherence as prior, norm, quad1, quad2, lin, and qmix.

judged relevant, i.e., blogs having a recurring interest in a given topic, are generated by blog models that generate blog posts with topical consistency. However, since we only see the posts collected during 11 weeks, the true topical distribution of the blog is only approximated by this observed sample.
Intuitively, in order to get a good estimation of the coherence of the underlying topical structure of the blog model, a certain number of posts should be contained in the sample under observation. Also, recall that in Section 3.2 on page 38 we have seen that the size of document sets has an impact on the calculation of coherence score: with same proportion of documents focusing on certain topic, larger document set generates a higher coherence score than smaller document set. This impact is especially significant when the document set is small, for example, less than 15 documents. From Table 4.2 in Section 4.2.2 we see that there exist many “small” blogs, e.g., blogs containing less than 10 posts.

This leads us to the following questions. What is the impact of the sample size on the estimation of the coherence of the true topical structure of the underlying blog model? Can we decide on a minimum number of posts to achieve a reliable estimation? And how would this threshold impact blog feed retrieval performance? Below, we address these questions with exploratory experiments.

### 4.6.1 Impact of sample size on the estimation of coherence

Intuitively, we expect that a larger sample will provide us with a better approximation of the true topic distribution of the population, i.e., a blog with more posts within the 11 week period of the data set should be a better approximation of the distribution of the topical structure of the blog in an infinite amount of time. Moreover, it is also intuitive that populations of different sizes require different minimum sample sizes for a reliable approximation. Since we do not know the size of the population, i.e., we do not know the number of posts a blog contains outside the 11 weeks covered by the data set, we need to decide on a minimum number of posts that would be sufficient for populations of different sizes.

To this end, we collect blogs with different numbers of posts from the Blog06 collection: blogs with at most 50 posts, 50–100 posts, 399–499 posts, 500–999 posts. For each number of these four groups, we sample 50 blogs for experiments.

For each blog \( B \) we collected, we calculate its coherence, which we denote as \( Co(B) \). We then sample a different number of posts: 5, 10, 20, 30, 40, and 50,\(^4\) and calculate the coherence score for each sample, which we denote as \( Co(S^k) \), where \( k = 5, 10, 20, 30, 40, 50 \) is the sample size. For each sample size, we generate 30 runs. We analyze how the value of \( Co(S^k) \) approximates the value of \( Co(B) \) as \( k \) changes by calculating the Mean Squared Error (MSE) of the sample coherence scores from the real coherence scores derived from the original blog using Eq. 4.15.

\[
MSE(Co(S^k)) = \frac{1}{n} \sum_i \left( Co(s^k_i) - Co(B) \right)^2, \tag{4.15}
\]

\(^4\)Note that this observation is made under simplified assumptions, i.e., the situation where the document sets can be divided into two self-coherent and mutually exclusive subsets. Nevertheless, it gives us sufficient motivation to check this phenomenon in practice.

\(^5\)For the set of blogs of 50 posts, we ignore the case of sampling 50 posts, as in this case it is equivalent to select all posts in a blog and therefore no approximation is needed.
where \( i = 1, \ldots, 30 \), \( S^k = \{s^k_i\}_{i=1}^{30} \) is the set of samples from the 30 runs, which are drawn from the original blog \( B \).

To summarize the trends of the impact of sample size on estimating the real coherence for a blog, we take the average MSE of the 50 blogs of different number of posts. Figure 4.5 shows the results. We see that as the sample size increases, the average MSE decreases. In particular, after 20 posts, the changes in average MSEs become very small compared to that before 20 posts. This trend applies to blogs with different numbers of posts, which suggests that no matter how large the actual size of the blog would be in an infinite amount of time, a minimum number of 20 posts can achieve a stable estimation of the true topical structure of a blog.

![Figure 4.5: Relation between the sample size and the average MSE of the sampled coherence score from the real coherence score. Here, len50, len100, len300, len500 denote the samples of blogs with 50, 100, 399–499, 500–999 blogs, respectively.](image)

**4.6.2 Relation between the population coherence and the accuracy of being approximated by sampled coherence**

One may notice that in Figure 4.5, for the same sample size, e.g., 5 posts sample, blogs with 500–999 posts have a lower MSE than blogs with 50 posts. This is counterintu-
Chapter 4. Blog Retrieval: Topical Consistency among Documents

<table>
<thead>
<tr>
<th>blogs of different sizes</th>
<th>50</th>
<th>100</th>
<th>300–499</th>
<th>500–999</th>
</tr>
</thead>
<tbody>
<tr>
<td>population coherence score</td>
<td>0.5344</td>
<td>0.5755</td>
<td>0.5091</td>
<td>0.7339</td>
</tr>
</tbody>
</table>

Table 4.4: The average coherence score of blogs with different number of posts.

itive. Indeed, we would expect that it is more difficult to approximate the distribution of a large population than a small one with the same amount of samples. In other words, we expect the average MSE of blogs with more than 500 posts to be higher than that of blogs with 50 posts. This unexpected phenomenon suggests that there are other factors besides the sample size that impact the estimation of the topical structure of the underlying blog model. A potential dimension is the coherence of the original blog, i.e., the coherence score of the population.

In Figure 4.6, we fix the sample size, and show the relation between the MSE of the sampled coherence score and the population coherence score. We see that the relation is non-linear, but there exists a pattern, which can be approximated by a quadratic function (shown in the plots). Particularly, if the population is extremely coherent, or extremely random, it has a better approximation.

In Table 4.4, we list the average coherence score of blogs with different numbers of posts that we used in the experiment discussed in Section 4.6.1. As we can see, the average population coherence score of blogs with more than 500 posts is much higher than that of blogs with 50 posts. This explains the phenomenon shown in Figure 4.5.

To wrap-up, the experiment in this section shows that for a given posts sample size, the coherence of the population is a factor that impacts the accuracy of the approximation. Populations with extremely random or extremely coherent topical structures are easier to be approximated. The relation between the population coherence and the accuracy of being estimated by sampled coherence is non-linear but does have a pattern (i.e., close to a quadratic relation).

4.6.3 Impact of sample size on blog feed retrieval

Exploring the impact of sampling size a step further, we experiment with post thresholds in the retrieval process. Blogs with fewer posts than the threshold are discarded from the results (both in the baseline setting, as well as in the coherence-based runs), leaving us with a thresholded blog feed retrieval runs. We use thresholds between 0 and 50 posts, and use the best performing parameter setting for the five models (i.e., $\sigma = .05$ for norm and $\gamma = .05$ for qmix). Figure 4.7 plots the relative increase in MAP for each of the models over the baseline for different thresholds.

From the plot we can conclude that the greatest relative improvement over the baseline occurs when only blogs with more than 20 posts are taken into consideration. The function norm is the only one to have its peak at a threshold of 30 posts. On the other hand, if the threshold eliminates too many blogs, the relative improvement will decrease since there may be very few relevant blogs left after thresholding. Table 4.5 lists the results for each of the functions and the baseline when using a threshold of
20 posts. The results show that in three cases the improvement over the baseline is significant (in terms of MAP), and that, again, the weighting function $q_{mix}$ performs best on all metrics.

The experiments in Sections 4.6.1 and 4.6.3 lead to the conclusion that coherence becomes beneficial for blogs when a blog contains more than 20 posts. This result suggests that it would be worth looking into the development of methods to estimate priors for blogs that are (currently) too short to derive benefit from the coherence score.

### 4.7 Conclusion

In this chapter we proposed a method to counteract the effects of topical noise in blogs with the goal of performing blog feed retrieval. For a blog to be relevant in a feed search task, it should show recurring interest in a given topic, something that is hard
Figure 4.7: Effect of threshold on difference in MAP between models and baseline.

<table>
<thead>
<tr>
<th>function</th>
<th>MAP</th>
<th>P@5</th>
<th>P@10</th>
<th>MRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>.2470</td>
<td>.4578</td>
<td>.4511</td>
<td>.6930</td>
</tr>
<tr>
<td>lin</td>
<td>.2533</td>
<td>.4756</td>
<td>.4689</td>
<td>.7174</td>
</tr>
<tr>
<td>norm</td>
<td>.2512</td>
<td>.4756</td>
<td>.4622</td>
<td>.7030</td>
</tr>
<tr>
<td>quad1</td>
<td>.2534</td>
<td>.4756</td>
<td>.4689</td>
<td>.7285</td>
</tr>
<tr>
<td>quad2</td>
<td>.2550</td>
<td>.4756</td>
<td>.4711</td>
<td>.7061</td>
</tr>
<tr>
<td>qmix</td>
<td>.2567</td>
<td>.4800</td>
<td>.4711</td>
<td>.7321</td>
</tr>
</tbody>
</table>

Table 4.5: Results of weighted coherence score applied to blogs with a minimum of 20 posts. Significance computed against the baseline. (Note that, compared to Table 4.3, the baseline has changed, due to the fact that blogs with fewer than 20 posts are eliminated from the collection.)

to measure due to the noisiness of blogs on a blog level. Within this context, we raise three research questions:

**RQ2a.** How do we measure topical consistency for a blog?

**RQ2b.** How can we use the coherence score in our blog retrieval process?

**RQ2c** Given that the collection we use in our experiments only provides us with a sample of blog posts generated by the underlying blog models, how does the sample size influence the estimation of the coherence and how does this influence blog feed retrieval?

For the first research question, we argued that established cohesion measures, in particular lexical cohesion calculated on the basis of lexical chains, are not suited for mea-
4.7. Conclusion

suring topical consistency in the blogsphere and proposed our coherence score which captures the topical clustering structure of a set of documents relative to a background collection. The coherence score can be calculated relatively efficiently. The calculation makes use of collection statistics only and requires neither external resources nor collection-specific parameter optimization. Applied to blogs, the coherence score reflects topical consistency, in other words, the level of topical noise of a blog.

With respect to research question RQ2b, we find that incorporating the coherence score in our retrieval framework required us to look at the relation between coherence and relevance. In case of a (topically) relevant blog, this blog should not be highly favored in the final ranking unless it is also topically coherent. On the other hand, blogs that have high topical coherence because they consistently treat a different topic than the given topic, should not enjoy unjustified promotion within the final ranking. To prevent this, we proposed weighting the coherence score by a notion of topical relevance. We compared two methods of incorporating the coherence: (i) a query-independent method, using coherence as prior, and (ii) a relevance-dependent method, where the coherence is weighted using a function of the retrieval score. Results show that the second method outperforms the baseline model, while the first method does not. Furthermore, the \textit{qmix} function performs best with significant improvement over the baseline on MAP and MRR, and non-significant improvements on the other metrics.

For research question RQ2c, following the intuition that the posts in our data set are a sample of the blogger’s posts, we expected a larger sample size to be a better approximation of the true distribution of posts. Our analysis of the relation between the sample size and the average deviation of the sampled coherence from the actual coherence of a blog shows that from 20 posts onwards this deviation does not change much anymore, indicating that 20 posts is the minimum sample size needed to get a proper estimation. This is further supported by blog feed retrieval experiments using only blogs that have more posts than a given threshold: using a threshold of 20 posts shows maximum relative improvement over the baseline.

We have shown the coherence score to be effective in capturing topical consistency in user generated content. Future work will focus on further optimization of the coherence score for use in blog feed retrieval, involving, for example, in-depth investigation of query-specific performance that could lead to further refinement of the weighting functions. An extension of the coherence score to other areas of user generated content, such as user reviews or audio blogs (podcasts) is a further avenue of future research.