Chapter 9

AN UNBIASED GENERATIVE MODEL FOR SETTING DISSEMINATION THRESHOLDS

Yi Zhang and Jamie Callan
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{yiz,callan}@cs.cmu.edu

Abstract Information filtering systems based on statistical retrieval models usually compute a numeric score that indicates how well each document matches each profile. Documents with scores above profile-specific dissemination thresholds are delivered. Optimal dissemination thresholds are usually difficult to determine a priori, so they are often learned during filtering, using relevance feedback about disseminated documents. However, the scores of disseminated documents are a biased sample of the complete distribution of document scores, which causes some algorithms to learn suboptimal thresholds.

This chapter presents a generative method of adjusting dissemination thresholds that explicitly models and compensates for this bias. The new algorithm, which is based on the Maximum Likelihood principle, jointly estimates the parameters of the density distributions for relevant and non-relevant documents and the ratio of relevant to non-relevant documents in the region around the dissemination threshold. Experiments demonstrate its effectiveness when its underlying assumptions about document scores are true, and illustrate its behavior when its assumptions don't match the actual distribution of document scores.

1. INTRODUCTION

Information filtering systems monitor a document stream to find documents that match information needs described by user profiles con-

sisting of queries and related context or history. Filtering systems based on statistical models (e.g., vector space, probabilistic, inference network, and statistical language models) use a numeric score (the *relevancy measure*) to indicate how well a document matches a profile, and only disseminate a document when its score is above some threshold (the *dissemination threshold*). As the information stream is processed, the system may be provided with relevance judgments for some of the documents it delivers. An *adaptive information filtering system* can learn from these user feedback so that it becomes more accurate over time.

A common approach to learning user profiles is an incremental version of the Rocchio algorithm (Rocchio, 1971; Allan, 1996; Callan, 1996):

$$Q^{'} = \alpha Q + \beta \frac{\sum_{d_i \in R} d_i}{\|R\|} - \gamma \frac{\sum_{d_i \in NR} d_i}{\|NR\|}$$

where Q is the initial profile vector, Q' is the new profile vector, R is the set of relevant documents, NR is the set of non-relevant documents, d_i is a document vector, and α, β, γ are constants indicating the relative value of each type of evidence. Machine learning algorithms for text classification have also been used for this task, for example, k-nearest neighbor, naive Bayes, statistical language modeling, and boosting (Schapire et al., 1998; Hull and Robertson, 2000; Robertson and Hull, 2001; Kraaij et al., 2000; Kim et al., 2000; Ault and Yang, 2001).

The problem of how to set filtering thresholds received little attention until the mid 1990s, in part because many operational environments relied upon Boolean queries and exact-match retrieval models. Researchers working with statistical retrieval algorithms typically delayed dissemination decisions while scores were computed for each document in the stream, sorted documents by their scores, and then disseminated the top N documents, as was done in the TREC Routing task (e.g., (Robertson et al., 1996)). This approach is effective when it is not necessary to disseminate information quickly. When dissemination decisions cannot be delayed, it is necessary to find dissemination thresholds for each profile. In the late 1990s dissemination thresholds were recognized as a crucial component of information filtering systems, and a variety of heuristic algorithms were developed (e.g., (Zhai et al., 1999; Zhai et al., 2000; Zhang and Callan, 2001b; Ault and Yang, 2001; Robertson and Walker, 2000; Robertson and Walker, 2001)).

This chapter focuses on using a generative model to dynamically determine dissemination thresholds while filtering. We assume that there is some filtering system, presumably based on a statistical model of information retrieval, that assigns a relevance-based score to each document. Our problem is to learn thresholds that determine whether to dissemi-

nate a document given its relevance score. We assume that a user provides relevance feedback about each disseminated document, enabling the system to adjust thresholds dynamically. In this chapter threshold quality is evaluated primarily by how well it optimizes a linear utility measure, as is common in the TREC Filtering Tracks (Hull and Robertson, 2000; Robertson and Hull, 2001; Robertson and Soboroff, 2002), but occasionally it is evaluated using other metrics.

The next section of this chapter introduces model based approaches to optimizing utility functions, using a Normal-Exponential model as an example. Section 3 discusses the sampling bias problem that occurs when using a generative model for threshold selection, and provides a solution that explicitly models the bias. Section 4 introduces our experimental methodology, including data sets, system settings, and evaluation metric. Section 5 presents experimental results that compare biased and unbiased versions of the algorithm on several different datasets; these experiments demonstrate the effectiveness of the algorithm when the model's underlying assumption is true, and the effects when the model itself is a poor fit to the actual distribution of document scores. Section 6 concludes.

2. GENERATIVE MODELS OF DISSEMINATION THRESHOLDS

The goal of setting a dissemination threshold for a filtering system is to satisfy the user. User satisfaction can be modeled by a utility function, as in recent TREC Filtering Track evaluations (Robertson and Hull, 2001; Hull and Robertson, 2000).

$$Utility = A \cdot R^+ + B \cdot N^+ + C \cdot R^- + D \cdot N^- \tag{9.1}$$

This model corresponds to assigning a positive or negative value to each element in the categories of Table 9.1, where R^-, R^+, N^-, and N^+ correspond to the number of documents that fall into the corresponding category, and A, B, C, and D correspond to the credit/penalty for each element in the category. For example, the TREC 2001 Filtering Track Utility was $T10U = 2R^+ - N^+$; the TREC 2002 Filtering Track used a normalized version of $T10U$ for evaluation.

Another commonly used evaluation metric is the F measure, which is a combination of Recall and Precision.

$$F = \frac{1}{\frac{a}{Recall} + \frac{b}{Precision}} \tag{9.2}$$

For example, the TREC 2001 and TREC 2002 Filtering Tracks also used $F = 1/(1/Recall + 4/Precision))$ for evaluation.

Table 9.1 The values assigned to relevant and non-relevant documents that the filtering system did and did not deliver.

	Relevant	*Non-Relevant*
Delivered	R^+/A	N^+/B
Not Delivered	R^-/C	N^-/D

The problem of setting dissemination thresholds is to find a threshold that maximizes the utility metric used to evaluate the filtering system. Let $P(R|d)$ represent the probability that document d is relevant to the profile. Maximizing the utility in Equation 9.1 is equivalent to delivering d to the user if and only if $A \cdot P(R|d) + B \cdot (1 - P(R|d)) > C \cdot P(R|d) + D \cdot (1 - P(R|d))$.[1] We express this requirement as shown below.

$$\text{Deliver } d \text{ if } P(R|d) > \frac{D - B}{A - B - C + B} \tag{9.3}$$

Unfortunately, most retrieval models provide only a score that is correlated (in some unknown way) with relevance, rather than the probability of relevance $P(R|d)$; this is true even for systems using probabilistic models. Our task is to transform the relevance score to $P(R|d)$. This task is similar to some other classification tasks where it is necessary to know the posterior $P(R|score)$. There are two major approaches to solving this problem in recent research.

Discriminant analysis: The posterior distribution $P(R|score)$ is modeled directly. For example, a research group from CLARITECH used a heuristic that allowed the threshold to vary between a lower bound utility value (zero) and an upper bound optimal value (Zhai et al., 2000). A group from Microsoft Cambridge mapped the score to the probability of relevance using a modified form of logistic regression (Robertson and Walker, 2001; Robertson and Walker, 2000).

Generative analysis: The distributions $P(score|R)$, $P(score|NR)$, $P(R)$, and $P(NR)$ are modeled. In this approach, once the model

[1] This is greedy optimization. In some cases, such as the early stage of filtering, it may be desirable to deliver even when $P(R|d)$ is smaller than required by this formula, e.g., to get more training data. There is little research on this topic and it is not discussed in this chapter.

is decided, $P(R|score)$ can be derived using Bayes rule.

$$\begin{aligned} P(R|score) &= \frac{P(R, score)}{P(score)} \\ &= \frac{P(score|R) \cdot P(R)}{P(score|R) \cdot P(R) + P(score|NR) \cdot P(NR)} \end{aligned}$$

For example, a research group from Katholieke Universiteit Nijmegan (KUN) assumed a Gaussian distribution for the scores of relevant documents and an Exponential distribution for the scores of non-relevant documents, and then found the threshold that maximized a given linear utility measure. This approach was extremely effective in one test, achieving the best result for utility oriented runs in the TREC-9 Filtering Track evaluation (Arampatzis et al., 2001).

Discriminative analysis approaches to modeling $P(R|score)$ tend to have lower bias and higher variance; generative analysis approaches usually have higher bias (because of the model assumptions), but lower variance (they use data more efficiently) (Hastie et al., 2001; Rubinstein and Hastie, 1997; Ng and Jordan., 2002).

In an information filtering environment, when the amount of training data is small, variance is a primary cause of error, thus generative approaches are a good choice. In addition to $P(R|score)$, generative analysis also estimates the marginal probability of relevance, which makes it possible to estimate Precision *and* Recall for a given threshold; discriminative models can estimate only Precision, because Recall can not be derived from only the conditional probability $P(R|score)$. The ability to estimate both Precision and Recall from generative models makes it possible to optimize for the F measure, if desired.

The ability of generative models to use training data efficiently, and their ability to optimize for either Utility or F measure, make them an attractive choice for learning dissemination thresholds. We restrict our attention to generative analysis in the remainder of this chapter.

2.1 NORMAL AND EXPONENTIAL MODELS OF DOCUMENT SCORE DISTRIBUTIONS

If the system had an accurate model of the distribution of document scores for the top ranking documents (relevant and non-relevant), the dissemination threshold could be set accurately. Several researchers suggest using a Gaussian distribution for modeling the scores of relevant documents and an Exponential distribution for modeling the scores

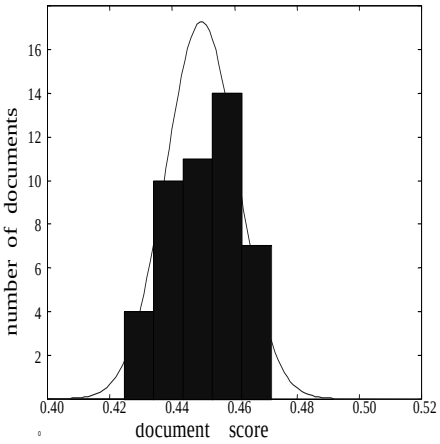

Figure 9.1 Density of relevant document scores for OHSU Topic 5. OHSUMED collection.

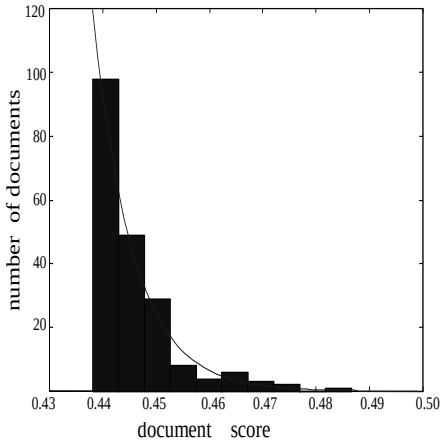

Figure 9.2 Density of non-relevant document scores for OHSU Topic 5. OHSUMED collection.

of the top ranking non-relevant documents (Arampatzis and Hameren, 2001; Manmatha et al., 2001). These distributions are modeled as:

$$P(score|R) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(score-\mu)^2}{2\sigma^2}} \tag{9.4}$$

$$P(score|NR) = \lambda e^{-\lambda(x-c)} \tag{9.5}$$

$$P(R) = p \tag{9.6}$$

where:
μ is the mean of the Gaussian distribution;
σ is the variance of Gaussian distribution;
λ is the variance of Exponential distribution;
c is the minimum score a non-relevant document can get; and
p is the the ratio of relevant documents to all documents in the region being modeled.

The parameter p does not represent the ratio in the corpus as a whole, because the Exponential model fits only the top non-relevant scores. The model is focused on, and is most accurate modeling, the scores of the top-ranking documents, where thresholds are typically set.

Analysis of experimental results on TREC-9 Filtering Track data support this approach. Figures 9.3 and 9.1 illustrate how the normal distribution fits relevant document scores for two TREC-9 topics. Figures

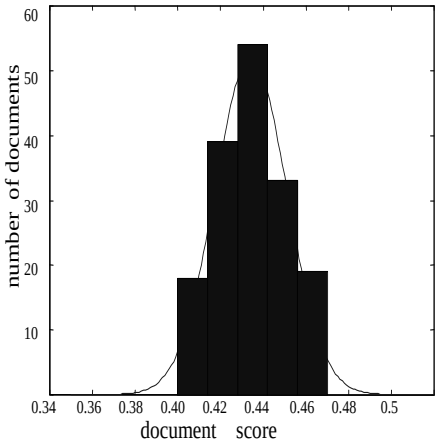

Figure 9.3 Density of relevant document scores for OHSU Topic 3. OHSUMED collection.

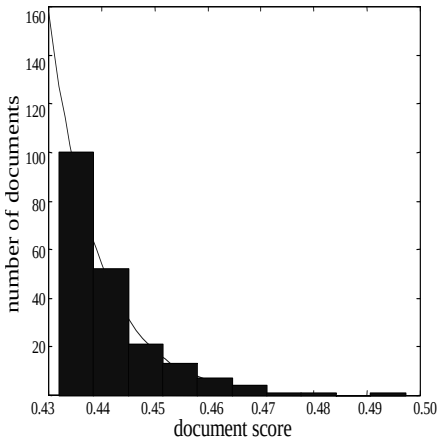

Figure 9.4 Density of non-relevant document scores for OHSU Topic 3. OHSUMED collection.

9.2 and 9.4 illustrate how the Exponential distribution fits the top 100 non-relevant document scores for the same topics.

This model is not perfect, especially for low scoring documents. However, it can provide a relatively accurate estimate of the distribution of scores for higher-scoring documents, and in the information-filtering task the area of interest (i.e., the area in which thresholds are likely to be located) usually is around the higher-scoring documents. A model that fits the higher-scoring documents well can be used for setting thresholds.

Based on this generative model for score distributions, we can calculate the probability of a document being relevant given its score.

$$
\begin{aligned}
P(R|score) &= \frac{P(score|R)P(R)}{P(score)} && (9.7)\\
&= \frac{P(score|R)P(R)}{P(score|R)P(R)+P(score|NR)P(NR)}\\
&= \frac{p\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(score-\mu)^2}{2\sigma^2}}}{p\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(score-\mu)^2}{2\sigma^2}}+(1-p)\lambda e^{-\lambda(score-c)}}\\
&= \frac{1}{1+\frac{(1-p)\lambda\sqrt{2\pi}\sigma}{p}e^{-\lambda(score-c)+\frac{(score-u)^2}{2\sigma^2}}} && (9.8)\\
&= \frac{1}{1+e^{c_1+c_2\cdot score+c_3\cdot score^2}} && (9.9)
\end{aligned}
$$

Equation 9.9 shows the relationship between generative and discriminative analysis: The discriminative function that corresponds to the Normal-Exponential model is a quadratic logistic regression. There are two options: Train a generative model to find the parameters $(\mu, \sigma, \lambda, p)$, or train a discriminative model to find the parameters (c_1, c_2, c_3). Prior research indicates that a generative model borrows strength from the marginal density and uses training data more efficiently, even when the goal is discriminative (Rubinstein and Hastie, 1997; Ng and Jordan., 2002). If our confidence in the generative model correctness is high, or if there is little training data, a generative model is a reasonable choice.

Using this model, filtering to optimize a linear utility function (Equation 9.3) is equivalent to setting the threshold th^* as shown below (Arampatzis and Hameren, 2001).

$$th^* = \begin{cases} (b - \sqrt{\Delta})/a & \text{if } \Delta \geq 0 \\ +\infty & \text{if } \Delta < 0 \end{cases} \tag{9.10}$$

where:

$$\begin{aligned} \triangle &= b^2 - a \cdot d \\ a &= \frac{1}{\sigma^2} \\ b &= \frac{\mu}{\sigma^2} + \lambda \\ d &= \frac{\mu^2}{\sigma^2} - 2\log\left(\frac{C-A}{B-D} \cdot p \cdot \frac{1}{\lambda\sqrt{2\pi}\sigma}\right) + 2\lambda \cdot c \end{aligned}$$

and A, B, C, and D are Utility function parameters (Equation 9.1).

For a given threshold th:

$$\begin{aligned} Recall &= pZ(\frac{th-\mu}{\sigma}) \\ Precision &= \frac{p(1 - Z(\frac{th-\mu}{\sigma}))}{p(1 - Z(\frac{th-\mu}{\sigma})) + (1-p)e^{-\lambda(th-c)}} \end{aligned}$$

where:

$$Z(\alpha) = \int_{-\infty}^{\alpha} \frac{1}{\sqrt{2\pi}} exp(-x^2/2)dx$$

Since the F measure is a function of Precision and Recall (Equation 9.2), optimizing for it is also straightforward once μ, λ, c and p are known.

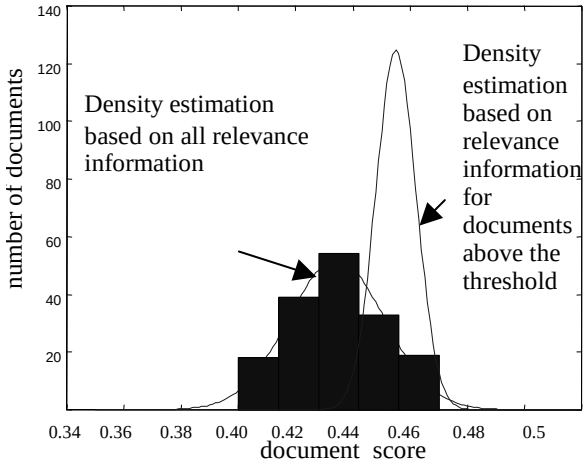

Figure 9.5 Estimation of parameters for the distribution of relevant document scores (OHSU Topic 3, OHSUMED collection). Parameters estimated without considering the sampling bias problem are overestimated.

3. THE NON-RANDOM SAMPLING PROBLEM & SOLUTION

Given a set of document scores, the basic parameter estimation method for normal and Exponential distributions is a simple calculation of the mean and variance over training data (Arampatzis and Hameren, 2001). One potential weakness with this simple approach is that it assumes that the training data accurately represents the distribution of relevant and non-relevant document scores. This assumption is not true in an adaptive filtering environment, because relevance information is obtained only for documents that are actually disseminated. Relevance information is not available for documents that have scores below the threshold, so the training data is inherently biased. We call this the *non-random sampling problem*. Estimation of $(\mu, \sigma, \lambda, p)$ without considering this problem is incorrect. If the user profile is static and training data is restricted to scores above the threshold, the mean of the sample scores is likely to be higher than the real mean.

For example, the true mean and variance of the Gaussian distribution for OHSU Topic 3 (based on all relevant documents) are 0.4343 and 0.0169 (Figure 9.3). However, if training data is restricted to documents with scores above a dissemination threshold of 0.4435, the learned mean and variance are 0.4551 and 0.007 (Figure 9.5). This difference can lead to an inaccurate dissemination threshold.

Non-random sampling is inherent in the training data used for setting dissemination thresholds. Any generative method that assumes randomly sampled training data (e.g., generative analysis methods mentioned in Section 2, not just the basic method presented here (Arampatzis and Hameren, 2001)), must solve this problem.

3.1 A GENERAL APPROACH TO UNBIASED ESTIMATION

In order to get an unbiased estimate of the distribution parameters, we must take into consideration the sampling constraint, which in a filtering environment is the dissemination threshold. In an *adaptive* filtering environment the threshold changes over time, and scoring function $s(d)$ may also change over time, so the problem becomes more complicated.

The non-random sampling problem is corrected by a new method of using generative analysis to set dissemination thresholds. Our new unbiased algorithm explicitly models the sampling bias, and uses the maximum likelihood principle, an unbiased parameter estimation method, to find unbiased estimates of the parameters. It jointly estimates the parameters of the two density distributions and the ratio of the relevant documents in the corpus (Zhang and Callan, 2001a).

At a certain point in the filtering process, the filtering system has already delivered N documents to a user and the user has provided relevance judgments for these documents (the training data). The i-th delivered document with user feedback is represented with a triple $(Feedback_i, Score_i, f_i)$, where:

$$Feedback_i = \begin{cases} R & \text{for a relevant document.} \\ NR & \text{for a non-relevant document.} \end{cases}$$

$Score_i$: The score of document d_i.

f_i : The function that delivered d_i. This contains information such as the relevance scoring function (s_i), and profile threshold (th_i) when d_i arrived. $f_i = s_i - th_i$.

s_i : The scoring function used to measure relevancy when d_i arrived. Usually $Score_i$ represents $s_i(d_i)$.

th_i : The threshold when d_i arrived.

Let θ represent the parameters for the generative model used to describe the density distribution of the scores. According to Bayes theorem, the most probable value of θ is:

$$\theta^* = \arg\max_{\theta} P(\theta|D)$$

$$= \arg\max_{\theta} \frac{P(D|\theta)P(\theta)}{P(D)}$$

For simplicity, we first assume that there is no prior knowledge of the distribution of θ and treat the prior probability of $P(\theta)$ as uniform.[2] Because $P(D)$ is a constant independent of H, it can be dropped. Thus the most probable θ is the one that maximizes the likelihood of the training data.

$$\begin{aligned}
\theta^* &= \arg\max_{\theta} P(D|\theta) \\
&= \arg\max_{\theta} \prod_{i=1..N} P(d_i|\theta) \\
&= \arg\max_{\theta} \sum_{i=1..N} \log P(d_i|\theta) \\
&= \arg\max_{\theta} \sum_{i=1..N} \log P(Score_i, Feedback_i|\theta, f_i(d_i) > 0) \quad (9.11)
\end{aligned}$$

The second step in Equation 9.11 is due to the assumption that each document is independent; the third step is due to the fact that maximizing a function is equivalent to maximizing its logarithm; and the last step indicates the sampling constraints for training data: $s_i(d_i) - th_i(d_i) > 0$. Notice that although the training data are not sampled randomly from the whole corpus, each individual training document is sampled randomly according to the conditional probability $P(Score = Score_i, R_i|\theta, f_i(d_i) > 0)$. So parameters estimated based on Equation 9.11 are unbiased.

3.2 UNBIASED ESTIMATION OF NORMAL AND EXPONENTIAL MODELS

If we assume a Gaussian distribution for the scores of relevant documents and an Exponential distribution for the scores of non-relevant documents as in (Arampatzis and Hameren, 2001), and if the profile scoring function is not changing over time (revisited in Section 5.3), then $\theta = (\mu, \sigma, \lambda, p)$ and $f_i(d_i) > 0 \Leftrightarrow Score_i > th_i$. (The meaning of $(\mu, \sigma, \lambda, p)$ was described in Section 2.1.)

For each item inside the sum operation of Equation 9.11, we have:

$$P(Score_i, Feedback_i|\theta, f_i(d_i) > 0)$$

[2]We will revisit and remove the assumption in Section 4.2 and use a conjugate prior of $P(\theta)$ for smoothing.

$$\begin{aligned} &= \frac{P(Score_i, Feedback_i, Score_i > th_i|\theta)}{P(Score_i > th_i|\theta)} \\ &= \frac{P(Score_i, Score_i > th_i|\theta, Feedback_i)P(Feedback_i|\theta)}{P(Score_i > th_i|\theta)} \\ &= \frac{P(Score_i|\theta, Feedback_i)P(Feedback_i|\theta)}{P(Score_i > th_i|\theta)} \end{aligned} \tag{9.12}$$

The first step in Equation 9.12 is based on the definition of conditional probability; the second step is due to the chain rule of probability; and the last step is due to the fact that all training documents must have a score higher than the threshold.

For convenience, let $f_1(\mu, \sigma, th_i)$ represent the probability of a document getting a score above threshold th_i if it is relevant and $f_2(\lambda, th_i)$ be the probability of it getting a score above threshold th_i if it is non-relevant. The probability of getting a score above th_i can be calculated by integrating the density function from th_i to positive infinity. That is:

$$\begin{aligned} f_1(\mu, \sigma, th_i) &= P(Score > th_i|R) \\ &= \int_{th_i}^{+\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}\, dx. \end{aligned} \tag{9.13}$$

$$\begin{aligned} f_2(\lambda, th_i) &= P(Score > th_i|NR) \\ &= \int_{th_i}^{+\infty} \lambda e^{-\lambda(x-c)}, dx \\ &= e^{-\lambda(th_i - c)} \end{aligned} \tag{9.14}$$

We use $g(\mu, \sigma, \lambda, p, th_i)$ to represent the probability of a document getting a score above threshold th.

$$\begin{aligned} g(\mu, \sigma, \lambda, p, th_i) &= P(Score > th_i|\theta) \\ &= P(R|\theta)P(Score > th_i|R, \theta) + \\ &\quad P(NR|\theta)P(Score > th_i|NR, \theta) \\ &= pf_1(\mu, \sigma, th_i) + (1-p)f_2(\lambda, th_i) \end{aligned} \tag{9.15}$$

An intuitive explanation of Equations 9.13, 9.14, and and 9.15 is illustrated in Figure 9.6. If we assume the sum of areas under the Exponential and Gaussian curves is 1, then $p \cdot f_1(\mu, \sigma, th_i)$ corresponds to the area to the right of and below the normal distribution curve. $(1-p)f_2(\lambda, th_i)$ corresponds to the area to the right of and below the Exponential distribution curve.

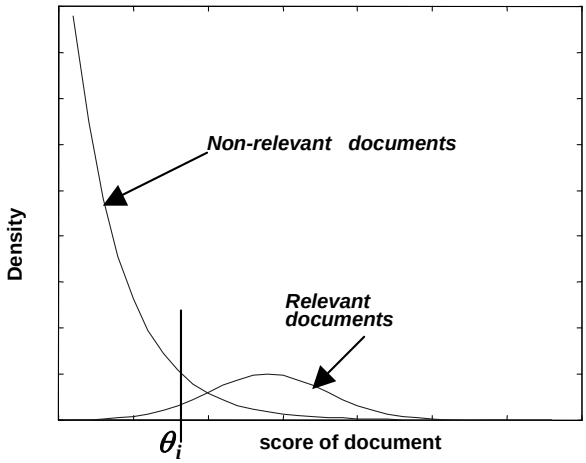

Figure 9.6 Distribution of scores of documents for a profile.

From the previous equations derived in this section:

$$(\mu^*, \sigma^*, \lambda^*, p^*) = \arg \max_{\mu,\sigma,\lambda,p} \sum_{i=1}^{N} LP_i \tag{9.16}$$

where for relevant documents:

$$LP_i = -\frac{(Score_i - u)^2}{2\sigma^2} + \log(p/(\sigma g(\mu, \sigma, \lambda, p, th_i)))$$

and for non-relevant documents:

$$LP_i = -\lambda(Score_i - c) + \log((1-p)\lambda/g(\mu, \sigma, \lambda, p, th_i))$$

If the training data are random samples of the whole corpus (not true for filtering), $th_i = -\infty$ and thus $g(\mu, \sigma, \lambda, p, th_i) = 1$. If $g(\mu, \sigma, \lambda, p, th_i)$ in Equation 9.16 is replaced with 1, the optimal parameters will be the same as those used in the basic, biased method (Arampatzis and Hameren, 2001). In other words, the basic, biased method is a maximum likelihood estimation of the parameters if the training data are a random sample of the dataset. However, $g(\mu, \sigma, \lambda, p, th_i)$ is not equal to 1 because the sample is biased during filtering. Equation 9.16 must be used to find the unbiased parameters.

There is no closed form solution for Equation 9.16, so numerical methods can be used. One such method is conjugate gradient descent (CGD). CGD is a gradient based optimization procedure for a function (Press

```
x = initial guess for the minimum
q = negative of gradient at x (search direction)
do {
   x  =  the minimal point along direction q
   q  =  a linear combination of new gradient and old q
} until converge
```

Figure 9.7 Outline of the Conjugate Gradient Descent algorithm.

et al., 1992; MacKay, 2001). A brief sketch of the CGD algorithm is shown in Figure 9.7. At each step, x is the approximate solution, q is the search direction, and x is improved by searching for a better solution along direction q. When using CGD to estimate parameters for Normal and Exponential models, the derivative of the right side of Equation 9.16 is used for calculating the search direction q.

4. EXPERIMENTAL METHODOLOGY

The biased and unbiased generative methods of setting dissemination thresholds were evaluated in a series of experiments with TREC data. The experimental methodology for the TREC Filtering Track changes from year to year. The experiments reported here used the TREC-9 experimental methodology (Robertson and Hull, 2001) in experiments with TREC-8, TREC-9, and TREC-10 Filtering data. The advantage of this approach is that experimental results obtained with different datasets can be compared, because they were obtained with the same experimental methodology. However, a consequence is that the results reported here for TREC-8 and TREC-10 data are not directly comparable to results from systems participating in the TREC-8 and TREC-10 Filtering Tracks. The evaluation measures, datasets, and experimental methodology are described in more detail below.

4.1 DATASETS

Three different text corpora were used in the experiments: the FT dataset used in the TREC-8 Filtering Track, the OHSUMED dataset used in the TREC-9 Filtering Track, and the Reuters 2001 dataset used in the TREC-10 Filtering Track. Each is discussed below.

4.1.1 FT Data. The FT data is a collection of 210,158 articles from the 1991 to 1994 Financial Times. It was used by the TREC-8 Filtering Track (Hull and Robertson, 2000). TREC topics 351-400 were

used to simulate user profiles. The relevance judgments were made by NIST on the basis of pooled output from several searches. For each profile, the system begins with two identified relevant documents and a natural language description of the information need, which is the title and description field of the corresponding TREC topic. The average number of relevant articles per profile in the testing data is 36.

4.1.2 OHSUMED Data. The OHSUMED data is a collection from the US National Library of Medicine's bibliographic database (Hersh et al., 1994). It was used by the TREC-9 Filtering Track (Robertson and Hull, 2001). It consists of 348,566 articles from a subset of 270 journals covering the years 1987 to 1991. 63 OHSUMED queries and 500 MeSH headings were used to simulate user profiles. The relevance judgments for the OHSUMED queries were made by medical librarians and physicians based on the results of interactive searches. For the MeSH headings, assignment of a heading to a document by the National Library of Medicine is treated as equivalent to a positive relevance judgment. In the TREC-9 Filtering Track, it is assumed that the user profile descriptions arrive at the beginning of 1988, so the 54,709 articles from 1987 can be used to learn word occurrence (e.g., idf) statistics and corpus statistics (e.g., average document length). For each user, the system begins with a natural language description of the information need and 2 examples of relevant documents. The average numbers of relevant articles per profile in the testing data are 51 for the OHSUMED topics and 249 for the MeSH headings.

4.1.3 Reuters 2001 Data. The Reuters 2001 data is a collection of about 810,000 Reuters English News stories from August 20, 1996 to August 19, 1997. It was used by the TREC-10 and TREC-11 Filtering Tracks (Robertson and Soboroff, 2002).

In TREC-10, 84 Reuters categories were used to simulate user profiles. For each profile, the system begins with two identified relevant documents and a natural language description of the information need, which is the title and description field of the corresponding topics provided by TREC-10. The average number of relevant articles in the testing data is about 9,795 documents per profile, which is much larger than the other two corpora.

4.2 FILTERING ENVIRONMENT

The YFilter information filtering system (Zhang and Callan, 2001b) was used for the experiments reported here. It processes documents by first removing symbols such as punctuation and special characters, ex-

cluding the 418 highly frequent terms listed in the default INQUERY stop words list (Broglio et al., 1995), and then stemming using the Porter stemmer (Porter, 1980). Processed documents are compared to each profile using the INQUERY variant of BM25 tf.idf formula (Robertson et al., 1996; Callan, 1996) to measure the similarity of the document and user profile. For each topic, the filtering system created initial filtering profiles using terms from the TREC topic Title and Description fields, and set the initial threshold to allow the highest-scoring d documents in the training dataset to pass. For simplicity, the filtering profile term weights were not updated while filtering. Because the first two relevant documents given to the system were not sampled under the constraint that their scores must exceed the dissemination threshold, their probabilities were simply $P(d_i|\theta) = P(Score_i, Feedback_i|\theta)$, and the corresponding element of Equation 9.16 was changed to:

$$LP_i = -\frac{(Score_i - u)^2}{2\sigma^2} - \log(\sigma)$$

In Section 3.2, for simplicity we set the prior probability to be uniform. However, in the real filtering task, especially during the early stage of filtering when there is only a small number of samples, this may cause problems. For example, if only non-relevant documents are delivered, the estimate of p will be 0 without a prior. Or, if all the relevant documents have the same score, the variance will be 0. Smoothing using a prior of parameters can solve these problems. The prior needn't be very accurate, because as the amount of sample data increases, the influence of the prior decreases. In our experiments, we set the prior of p as a beta distribution[3]: $p^{\varepsilon_1} \cdot (1-p)^{\varepsilon_2}$, which is equal to adding ε_1 relevant documents and ε_2 non-relevant documents sampled randomly for smoothing. The prior of σ^2 is set to be $e^{-\nu^2/(s\sigma^2)}$, which is equal to adding ν^2 to the sum of the square of the variance of relevant documents.[4] The value of $\varepsilon_1, \varepsilon_2$ and υ needn't to be very accurate and should be very small in order not to influence the final results. $\varepsilon_1, \varepsilon_2$ and υ were all set around 0.001-0.01 in our experiments.

4.3 METRICS

The effectiveness of the biased and unbiased generative methods of determining dissemination thresholds was measured using the linear utility

[3] Because the beta distribution is a conjugate prior for the binomial distribution, we use it as a prior to simplify the calculations.

[4] This is a special case of inverse gamma distribution, which is the conjugate prior of the variance of the normal distribution.

	Biased (Run1)	*Biased + Min. Delivery (Run2)*	*Unbiased (Run3)*	*Unbiased + Min. Delivery (Run4)*
T9$U^{'}$	1.84	3.25	2.70	8.17
Delivered docs per profile	3.83	9.65	5.73	18.40
Precision	0.37	0.29	0.36	0.32
Recall	0.04	0.08	0.05	0.14

Table 9.2 Comparison of the basic biased and unbiased Maximum Likelihood algorithms on the TREC-9 Filtering data. OHSUMED data, OHSU topics.

function described in Section 1 with $(A, B, C, D) = (2, 1, 0, 0)$. The resulting utility measure is:

$$T9U^{'} = 2R^{+} - N^{+}$$

which is a slightly simplified variant of the T9U utility metric used in the TREC-9 Filtering Track. R^{+} and N^{-} are the numbers of relevant and non-relevant documents disseminated. The corresponding delivery rule is:

$$\text{deliver if } P(Relevant|Score) > 0.33 \tag{9.17}$$

Experiments were also conducted with other values of (A, B, C, D), to test the sensitivity of each method to specific parameter settings.

5. EXPERIMENTAL RESULTS

Three sets of experiments were conducted. The first set investigated the effectiveness of the biased and unbiased generative methods of determining dissemination thresholds when the target is a specific utility metric. The experiments also studied the effect of including a minimum delivery constraint. The second set of experiments investigated the two approaches using a wider range of utility metrics; the goal of these experiments was to identify any sensitivity to the evaluation metric. A third set of experiments investigated behavior when the underlying assumption about the distributions of relevant and non-relevant document scores is violated. These three sets of experiments are described below.

5.1 BIASED VS. UNBIASED PARAMETER ESTIMATION

The first set of experiments investigated how well each method selects thresholds that optimize the $T9U^{'}$ Utility metric, which is consistent

	Biased (Run1)	*Biased + Min. Delivery (Run2)*	*Unbiased (Run3)*	*Unbiased + Min. Delivery (Run4)*
T9U'	1.89	4.28	2.44	13.10
Delivered docs per profile	3.51	11.82	6.22	27.91
Precision	0.42	0.39	0.40	0.34
Recall	0.02	0.05	0.03	0.07

Table 9.3 Comparison of the basic biased and unbiased Maximum Likelihood algorithms on the TREC-9 Filtering data. OHSUMED data, MeSH topics.

with recent TREC Filtering Track evaluations. Optimizing Utility can lead to higher Precision or Recall as a side-effect, but is not guaranteed to do so. In some cases the best way to increase Utility is to increase Precision (higher threshold); in other cases the best way to increase Utility is to increase Recall (lower threshold). All three metrics are reported, to provide greater insight into the experimental results, but Utility, the metric being optimized, is the focus of the discussion.

Experiments were conducted on TREC-8 and TREC-9 Filtering Track data. Four runs were conducted using each dataset. The basic *biased* parameter estimation method (Arampatzis et al., 2001) served as the baseline method ("Run 1"). The *unbiased* run ("Run 3") used the unbiased maximum likelihood estimation. Both runs stopped delivering documents when Δ was negative.

It is especially common for Δ to be negative at the beginning of filtering; in these cases, no documents are disseminated, no user feedback is available, and no learning occurs. A minimum delivery constraint was introduced to avoid this problem. If a profile had not achieved the minimum delivery constraint, its threshold was decreased automatically. Runs 2 and 4 correspond to the biased and unbiased algorithms using a minimum delivery constraint that required an average of 10 documents to be disseminated to each profile.[5]

Neither algorithm worked well on the OHSUMED dataset without a minimum delivery constraint (Tables 9.2 and 9.3, columns 1 and 3). Especially at the early stage of filtering, with only about 2 documents delivered, it is very hard to estimate the score distribution correctly.

[5] The goal is to restart profiles that stopped disseminating because their thresholds were mistakenly set too high, so the minimum delivery constraint does not need to be large. The 10 document constraint was chosen arbitrarily.

	Biased (Run1)	*Biased + Min. Delivery (Run2)*	*Unbiased (Run3)*	*Unbiased + Min. Delivery (Run4)*
T9U'	1.44	-0.200	0.65	0.84
Delivered docs per profile	9.58	10.44	9.05	12.27
Precision	0.20	0.17	0.22	0.26
Recall	0.16	0.17	0.15	0.19
Profiles with Precision=0.0	24	22	24	10

Table 9.4 Comparison of the basic biased and unbiased Maximum Likelihood algorithms on TREC-8 Filtering data.

When $\Delta < 0$ in Equation 9.9, the threshold is set too high to let any future documents be delivered, hence no learning takes place occurs. Introducing a minimum delivery constraint (i.e., forcing the system to deliver at least a certain percentage of documents) helps the system to recover automatically (Tables 9.2 and 9.3).

On the OHSUMED dataset, for both OHSU topics and MeSH topics, the unbiased maximum likelihood estimation plus a minimum delivery constraint achieved the best result. Although profile updating was disabled while filtering, all of the runs on this dataset received a positive average utility. The results for Run 4 on OHSU topics were above average compared with other filtering systems in the TREC-9 Adaptive Filtering Track (Robertson and Hull, 2001). This result indicates how effective the threshold-setting algorithm is, because the other filtering systems learned improved profiles while filtering, whereas all of our experiments used static filtering profiles.

All four algorithms performed equally well on FT data (Table 9.4). One difference between the FT dataset and the OHSUMED dataset is the average number of relevant documents per profile in the testing set. Most of the FT filtering profiles are not good profiles, which means it is almost impossible to find a threshold that achieves a positive utility without profile updating. Indeed, the biased method (Run 2) set thresholds so high for some profiles that no relevant documents were delivered, thus getting zero Precision and Recall on those profiles. The unbiased algorithm (Run 4) does not increase the threshold much, which is why the biased and unbiased methods appear similar according to the utility metric yet different according to the Precision and Recall metrics.

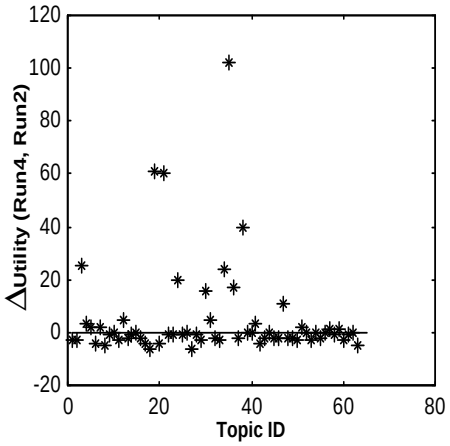

Figure 9.8 The improvement in utility of the unbiased method (Run 4) as compared with the biased method (Run 2). A minimum delivery constraint is used for both runs. Similar effectiveness is indicated by points near the horizontal line.

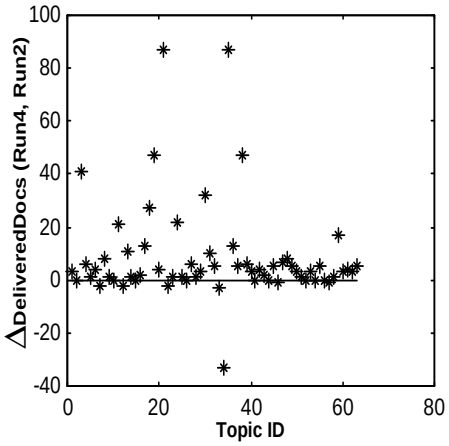

Figure 9.9 The improvement in the number of delivered documents for the unbiased method (Run 4) as compared with the biased method (Run 2). A minimum delivery constraint is used for both runs. Similar dissemination levels is indicated by points near the horizontal line.

Indeed, this experiment demonstrates quite clearly that optimizing for utility is not guaranteed to improve Precision or Recall.

When we compared the utilities of each OHSU topic on Run 4 and Run 2 (Table 9.2), we found that the unbiased method are more likely to do well on some profiles where the score distribution of relevant documents and non-relevant documents match the model's assumptions. For other profiles, the difference between the unbiased ML method and the basic, biased algorithm is not large. Comparison at the topic level and comparison between the OHSUMED and FT corpora confirms this conclusion (Figure 9.8).

The biased method delivered fewer OHSUMED documents, on average, than the unbiased ML method (Figure 9.9, and Tables 9.2, 9.3, and 9.4, columns 2 and 4). The average number of documents delivered by the biased method was close to the minimum delivery constraint, which is empirical justification for our previous analysis illustrated in Figure 9.5: The Gaussian mean estimated by the biased method was higher than the actual mean, and the corresponding result was that the thresh-

	Biased Estimation			*Unbiased Estimation*			Δ
(A, B)	*Utility*	*Precision*	*Recall*	*Utility*	*Precision*	*Recall*	*Utility*
(1,-1)	-3.09	0.22	0.16	-3.19	0.22	0.15	-3%
(2,-1)	-2.65	0.20	0.22	-2.58	0.20	0.20	$+3\%$
(3,-1)	-1.65	0.18	0.24	-0.42	0.19	0.24	$+75\%$
(4,-1)	2.19	0.16	0.27	5.60	0.18	0.26	$+156\%$
(8,-1)	32.47	0.15	0.30	32.35	0.14	0.35	0%
(16,-1)	115.19	0.11	0.44	117.35	0.13	0.40	$+2\%$

Table 9.5 Comparison of the basic biased and unbiased Maximum Likelihood algorithms on TREC-8 Filtering data, using different Utility metrics. FT data and topics. C=0 and D=0.

old was set too high.[6] The final results for the biased method appear to be highly influenced by the minimum delivery constraint. This is less a problem for the unbiased maximum likelihood method, because thresholds based on unbiased estimates of score distributions are more accurate.

We have not focused on computational efficiency in this chapter, but computational efficiency is important in environments where the filtering system must keep pace with a high-speed document stream. Although the mathematics may look complex, the unbiased ML algorithm is computationally efficient. It took about 21 minutes ("wall-clock" time) to filter 4 years of OHSUMED data for 64 OHSU profiles (Table 9.2, Run 4) on a 500 MHz Intel Pentium III processor with 256 MB of memory. This time includes all aspects of document filtering, including the setting and updating of dissemination thresholds while filtering.

5.2 VARYING THE UTILITY METRIC

The biased and unbiased algorithms can be compared using different utility metrics by varying the values of (A, B, C, D) in the linear utility function (Equation 9.1). A second series of experiments was conducted that examined effectiveness when A was varied between 1 and 16, which corresponds to delivery rules "deliver if $P(Relevance|Score) > p$" with

[6] Theoretically the threshold set by the biased algorithm could be higher or lower than optimal. The sample bias problem will make the estimation of both the negative mean and the positive mean biased high, and the final threshold depends on both.

	Biased Estimation			*Unbiased Estimation*			Δ
(A, B)	*Utility*	*Precision*	*Recall*	*Utility*	*Precision*	*Recall*	*Utility*
(1,-1)	-1.40	0.27	0.11	-1.75	0.27	0.11	+25%
(2,-1)	3.55	0.26	0.18	6.38	0.27	0.17	+80%
(3,-1)	12.54	0.25	0.22	17.00	0.26	0.20	+36%
(4,-1)	25.70	0.24	0.24	29.63	0.25	0.22	+15%
(8,-1)	100.65	0.22	0.34	98.62	0.23	0.28	−2%
(16,-1)	287.71	0.18	0.42	285.83	0.20	0.38	−1%

Table 9.6 Comparison of the basic biased and unbiased Maximum Likelihood algorithms on TREC-9 Filtering data, using different Utility metrics. OHSUMED data and OHSU topics. C=0 and D=0.

p varying from 50% ($A = 1$) to 5.9% ($A = 16$). Tables 9.5 and 9.6 summarize the experimental results.[7]

As the filtering goal was varied from "high Precision" (small A) to "high Recall" (large A), the value of unbiased estimation varied, too. The two algorithms were about equally effective when $A = 1$ ("high Precision") and $A \geq 8$ ("high Recall"). The unbiased Maximum Likelihood algorithm generally outperformed the basic, biased algorithm when $2 \leq A \leq 4$ ("medium Precision").

These results match our expectations. The filtering profiles were not sufficiently accurate for any dissemination threshold to satisfy the "high Precision" goal, leading to slightly negative utilities for both algorithms. The profiles were accurate enough to satisfy the "medium Precision" goal, and the unbiased algorithm generally made better use of its training data in these cases. Biased sampling is not a serious problem when dissemination thresholds are set low, as occurs when the goal is high Recall; in this case the biased and unbiased algorithms produced similar results.

5.3 BEHAVIOR WHEN ASSUMPTIONS ARE VIOLATED

This chapter presents a method of using unbiased generative analysis to set dissemination thresholds. Generative analysis makes strong as-

[7] The results for $A = 2$ differ slightly from those reported in the previous section (Tables 9.2 and 9.4) because of changes to the software between the two experiments. The changes were primarily in how thresholds were lowered for profiles that did not satisfy the minimum delivery constraint, and in how thresholds were set when there was insufficient data to apply the generative method, e.g., during the early stage of filtering.

	Biased	*Unbiased*	*Best System*
T9U'	2458	2759	6799
T10SU	0.135	0.143	0.291
Precision	0.505	0.505	0.538
Recall	0.140	0.158	0.496

Table 9.7 Comparison of the basic biased algorithm, the unbiased Maximum Likelihood algorithm, and the best system in the TREC-10 evaluation. Reuters 2001 data and Reuters 2001 category profiles.

sumptions about the underlying document score distributions, and these assumptions allow the algorithm to use training data very efficiently to discover the model parameters. However, when the model assumptions are wrong, i.e., when the actual distribution of document scores does not match the assumed distribution, the algorithm's behavior is undefined.

Two research groups participating in the TREC-10 Filtering track set dissemination thresholds using generative methods based on Normal and Exponential models (Arampatzis, 2002; Zhang and Callan, 2002). The utility scores were disappointing in both cases. Table 10 summarizes the TREC-10 results from Zhang and Callan, 2002 ("biased" and "unbiased") and from the best system participating in the track.[8,9] The weak results for systems using generative models suggests that there might be problems with that approach to setting dissemination thresholds.

Closer inspection of the results revealed that the model assumption was wrong for some profiles, i.e., the distributions of relevant and/or non-relevant document scores did not match Gaussian and/or Exponential distributions. Two examples illustrate how the models failed to match the distribution of document scores, and suggest possible solutions.

Reuters 2001 profile 77 had a low utility score, so it was a candidate for deeper analysis. We used the final version of the learned profile (presumably a high quality set of query terms and weights) to re-score all of the documents in the corpus. Figure 9.10 shows the score density distribution of the relevant documents, which can be approximated by a normal distribution. Figure 9.11 shows the score density distribution for non-relevant documents that contain at least one profile term.

[8]The best TREC-10 Filtering track result was run oraAU082201, which was submitted by Oracle (Robertson and Soboroff, 2002).

[9]TREC-10 used a scaled utility metric, defined as $T10SU = \frac{max(T9U, MinU) - MinU}{MaxU - MinU}$, where $MinU = -100$, $MaxU = 2 \cdot \text{TotalRelevant}$.

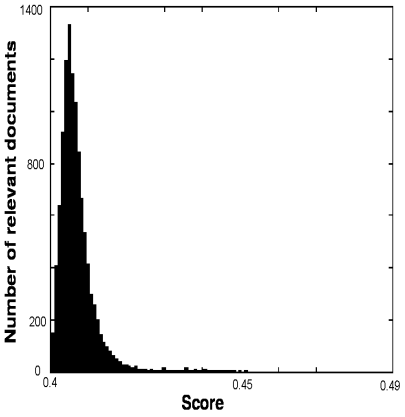

Figure 9.10 Score density distribution of relevant documents for profile 77. Reuters 2001 data.

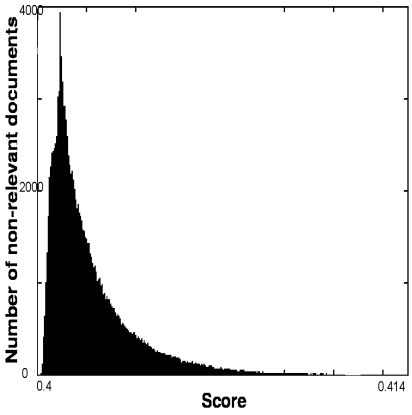

Figure 9.11 Score density distribution of non-relevant documents for profile 77. Reuters 2001 data.

An Exponential model is not a completely accurate model of the probability density function of non-relevant documents for this profile. The inaccuracy is not a problem if the filtering goal is high or medium Precision because the training data is from the right side of the distribution, which matches an Exponential model. If the filtering goal is high Recall, which includes training data from the left side of the distribution, a beta distribution would be a better fit. Since the Exponential distribution is a special case of the beta distribution, using a beta distribution would also cover cases where the Exponential distribution is the correct choice.

The maximum likelihood estimation method proposed in Equation 9.11 does not require any specific distribution; the beta distribution can be inserted into the general framework and the optimal parameters can be found. We did not implement the algorithm to find the optimal beta distribution parameters; we simply observe that it may be a better approximation function than the Exponential distribution proposed by (Arampatzis and Hameren, 2001; Manmatha et al., 2001) and used in our experiments. In fact this observation suggests that the type of model to use depends on the specific dataset and scoring algorithm. A more complicated model, such as the beta distribution, is likely to have less bias (making it more widely applicable) but more variance (requiring more training data).

Most of the TREC-10 filtering systems did poorly on Reuters 2001 profile 71, so it too was a candidate for deeper analysis. We used the final version of the learned profile (presumably a high quality set of query

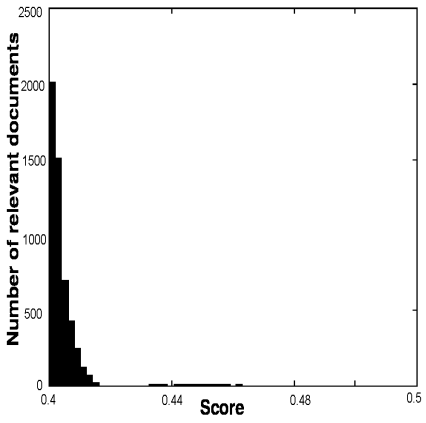

Figure 9.12 Score density distribution of relevant documents for profile 71. Reuters 2001 data.

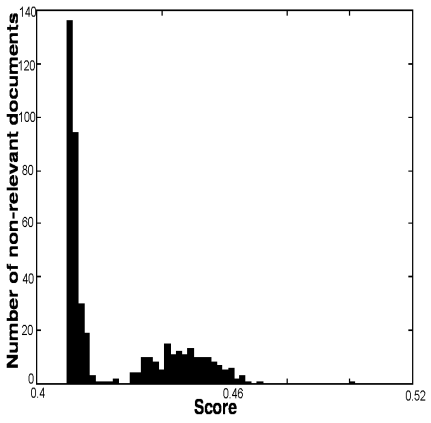

Figure 9.13 Score density distribution of high score relevant documents for profile 71. Reuters 2001 data.

terms and weights) to re-score all of the documents in the corpus. Figure 9.12 shows the score density distribution for relevant documents.

The score density distribution for relevant documents looks initially more like an Exponential distribution than a Gaussian distribution (Figure 9.12). However, "zooming in" on the score density distribution of the top scoring relevant documents shows that the distribution actually looks similar to a mixture of Exponential and Gaussian distributions (Figure 9.13).

The non-random sampling problem was identified in Section 3. as a cause of error in learning dissemination thresholds. However, it is also a cause of error in learning profiles, i.e., the terms and term weights that determine the score of a particular document. In our experiments, and indeed in most adaptive filtering research, the only relevant documents available to the profile learning algorithm are those that were disseminated. Thus the training data is a biased sample of the relevant documents.

The score density distribution of profile 71 is an extreme but real case that illustrates this problem. The information need described by profile 71 is a broad topic. With only two seed documents for training, the system tends to focus on a local area near the seed documents. The final learned profile, which is the function used to determine document scores, gives those documents and their close neighbors very high scores (corresponding to the normal area in Figure 9.13). The other relevant documents look to the profile just like non-relevant documents, hence

they tend to get low scores (corresponding to the Exponential area in Figures 9.12 and 9.13).

Although this problem affects the profile-learning algorithm (e.g., Rocchio), the effect on the threshold-learning algorithm is very strong in some cases, as illustrated by this example.

This chapter proposed an algorithm to solve the sampling bias problem for setting dissemination thresholds *in isolation from the rest of the filtering system.* It didn't develop a solution to solve the sampling bias problem when terms, term weights, and thresholds are all being adjusted simultaneously. However, the bias problem for threshold learning and profile term updating are correlated and should really be solved together. One solution is to explicitly model the sampling bias while profile term weights and threshold are changing. One could begin with Equation 9.11 and model

$$P(Score_i, Feedback_i|\theta, f_i(d_i) > 0)$$

The solution would depend on what kind of scoring function and profile learning algorithm the filtering system uses. Another possible solution is to deliver interesting "near miss" documents, so that the learning software gets a broader view of the surrounding information landscape, thereby learning a less biased scoring function. Theories from other research areas, such as active learning and reinforcement learning, are also potentially useful considering the similarity of the tasks.

6. CONCLUSION

Generative modeling has received considerable attention recently. This chapter describes how it can be used to set dissemination thresholds for adaptive information filtering systems. Generative models embody strong assumptions about the distributions of relevant and non-relevant document scores, which enables an adaptive filtering system to learn dissemination thresholds from small amounts of training data.

A major problem for generative methods is the non-random sampling that affects adaptive filtering systems when the only source of training data is the set of documents the system disseminates. Non-random sampling causes the system to overestimate the mean scores of relevant and non-relevant documents, and to underestimate the variance. These systematic errors can produce suboptimal dissemination thresholds, especially when the goal is high Precision output.

The non-random sampling problem can be corrected by explicitly modeling and compensating for the bias inherent in the filtering environment, where training data consists only of documents with scores above a threshold that changes constantly. This chapter describes a gen-

eral framework based on the maximum likelihood principle to estimate model parameters, using Normal and Exponential distributions as an example. An explicit solution based on this framework is provided to jointly estimate (i) the parameters of the density distributions for relevant and non-relevant document scores, and (ii) the ratios of relevant and non-relevant documents around the dissemination threshold. We believe this is the first research to explicitly model and compensate for the sample bias problem in an information filtering environment.

Non-random sampling is inherent in the adaptive filtering task scenario used in the TREC Filtering track evaluations, and it affects profile-learning as well as threshold-setting. It is common to treat the learning of profiles and dissemination thresholds as separate problems, but they are in fact highly related. Ignoring non-random sampling when learning profiles makes it more likely that the resulting document score distributions will be difficult for a threshold-setting algorithm to model.

The unbiased generative method of setting dissemination thresholds appears to offer the most advantage over alternatives when the filtering goal is "medium-to-high Precision". In these situations there is usually only a small amount of positive training data, and it is a very biased sample of the underlying distribution, so it is important to use an algorithm that compensates for the bias and uses training data efficiently. When the filtering goal is "high Recall", more training data is available, and it is a less biased sample, so the unbiased algorithm has less of an advantage.

The experimental results in this chapter demonstrate the strengths and weaknesses of using strong statistical models to set dissemination thresholds. When the model assumptions match the data, the unbiased generative method is quite effective; otherwise, its behavior is unpredictable. The unbiased generative framework presented here is more general than the Gaussian and Exponential models that were the focus of the chapter; a wide range of other models could be supported. Which models would be most general or most effective, or how one might determine during filtering which models best fit the data, are interesting topics for future research.

Acknowledgments

We thank Avi Arampatzis, R. Manmatha, Chengxiang Zhai, Wei Xu and Tom Minka for valuable discussions on some of the work in this chapter.

This material is based in part on work supported by Air Force Research Laboratory contract F30602-98-C-0110. Any opinions, findings, conclusions or recommendations expressed in this chapter are the authors', and do not necessarily reflect those of the sponsors.

References

Allan, J. (1996). Incremental relevance feedback for information filtering. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 270–278.

Arampatzis, A. (2002). Unbiased S-D threshold optimization, initial query degradation, decay, and incrementality for adaptive document filtering. In *Proceeding of the Tenth Text REtrieval Conference (TREC-10)*, pages 596–603. National Institute of Standards and Technology, special publication 500-250.

Arampatzis, A., Beney, J., Koster, C., and van der Weide., T. (2001). Incrementality, decay, and threshold optimization for adaptive filtering systems. In *Proceeding of Ninth Text REtrieval Conference (TREC-9)*, pages 589–600. National Institute of Standards and Technology, special publication 500-249.

Arampatzis, A. and Hameren, A. (2001). The score-distribution threshold optimization for adaptive binary classification task. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 285–293.

Ault, T. and Yang, Y. (2001). kNN at TREC-9: A failure analysis. In *Proceeding of Ninth Text REtrieval Conference (TREC-9)*, pages 127–134. National Institute of Standards and Technology, special publication 500-249.

Broglio, J., Callan, J., Croft, W., and Nachbar, D. (1995). Document retrieval and routing using the INQUERY system. In *Proceeding of Third Text REtrieval Conference (TREC-3)*, pages 29–38. National Institute of Standards and Technology,special publication 500-225.

Callan, J. (1996). Document filtering with inference networks. In *Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 262–269.

Hastie, T., Tibshirani, R., and Friedman, J. (2001). *Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer-Verlag.

Hersh, W., Buckley, C., J.Leone, T., and Hickam, D. (1994). OHSUMED: An interactive retrieval evaluation and new large test collection for research. In *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 192–201.

Hull, D. A. and Robertson, S. (2000). The TREC-8 Filtering track final report. In *Proceeding of the Eighth Text REtrieval Conference*

(TREC-8), pages 35–56. National Institute of Standards and Technology, special publication 500-246.

Kim, Y., Hahn, S., and Zhang, B. (2000). Text filtering by boosting Naive Bayes classifiers. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 168–175. ACM Press.

Kraaij, W., Pohlmann, R., and Hiemstra, D. (2000). Twenty-One at TREC-8: Using language technology for information retrieval. In *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, pages 285–300. National Institute of Standards and Technology, special publication 500-246.

MacKay, D. J. (2001). Macopt – a nippy wee optimizer. http://wol.ra.phy.cam.ac.uk/mackay/c/macopt.html.

Manmatha, R., Rath, T., and Feng, F. (2001). Modeling score distributions for combining the outputs of search engines. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 267–275.

Ng, A. Y. and Jordan., M. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and Naive Bayes. In *Proceeding of Fourteenth Neural Information Processing Systems.*

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, 14(3):130–137.

Press, W., Teukolsky, S., Vetterling, W., and Flannery, B. (1992). *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press.

Robertson, S. and Hull, D. (2001). The TREC-9 Filtering track report. In *The Ninth Text REtrieval Conference (TREC-9)*, pages 25–40. National Institute of Standards and Technology, special publication 500-249.

Robertson, S. and Soboroff, I. (2002). The TREC-10 Filtering track final report. In *Proceeding of the Tenth Text REtrieval Conference (TREC-10)*, pages 26–37. National Institute of Standards and Technology, special publication 500-250.

Robertson, S. and Walker, S. (2000). Threshold setting in adaptive filtering. *Journal of Documentation*, pages 312–331.

Robertson, S. and Walker, S. (2001). Microsoft Cambridge at TREC-9: Filtering track. In *Proceeding of Ninth Text REtrieval Conference (TREC-9)*, pages 361–368. National Institute of Standards and Technology, special publication 500-249.

Robertson, S., Walker, S., Beaulieu, M. M., Gatford, M., and Payne, A. (1996). Okapi at TREC-4. In *Proceeding of Fourth Text REtrieval*

Conference (TREC-4), pages 73–96. National Institute of Standards and Technology, special publication 500-236.

Rocchio, J. J. (1971). Relevance feedback in information retrieval. In *The SMART Retrieval System– Experiments in Automatic Document Processing*, pages 313–323. Prentice Hall.

Rubinstein, Y. D. and Hastie, T. (1997). Discriminative vs informative learning. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, pages 49–53.

Schapire, R., Singer, Y., and Singhal, A. (1998). Boosting and Rocchio applied to text filtering. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 215–213.

Zhai, C., Jansen, P., Roma, N., Stoica, E., and Evans, D. (2000). Optimization in CLARIT adaptive filtering. In *Proceeding of Eighth Text REtrieval Conference (TREC-8)*, pages 253–258. National Institute of Standards and Technology 500-246.

Zhai, C., Jansen, P., and Stoica, E. (1999). Threshold calibration in CLARIT adaptive filtering. In *Proceeding of Seventh Text REtrieval Conference (TREC-7)*, pages 149–157. National Institute of Standards and Technology, special publication 500-242.

Zhang, Y. and Callan, J. (2001a). Maximum likelihood estimation for filtering thresholds. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 294–302.

Zhang, Y. and Callan, J. (2001b). Yfilter at TREC-9. In *Proceedings of the Ninth Text REtrieval Conference (TREC-9)*, pages 135–140. National Institute of Standards and Technology, special publication 500-249.

Zhang, Y. and Callan, J. (2002). The bias problem and language models in adaptive filtering. In *The Tenth Text REtrieval Conference (TREC-10)*, pages 78–83. National Institute of Standards and Technology, special publication 500-250.