Dirichlet Mixtures for Query Estimation in Information Retrieval

Mark D. Smucker
David Kulp
James Allan

April 2005

Center for Intelligent Information Retrieval, Department of Computer Science, University of Massachusetts, Amherst, CIIR Technical Report IR-445

Abstract

Treated as small samples of text, user queries require smoothing to better estimate the probabilities of their true model. Traditional techniques to perform this smoothing include automatic query expansion and local feedback. This paper applies the bioinformatics smoothing technique, Dirichlet mixtures, to the task of query estimation. We discuss Dirichlet mixtures' relation to relevance models, probabilistic latent semantic indexing, and other information retrieval techniques. We describe how Dirichlet mixtures give insight into the value of retaining the original query in query expansion. On the task of adhoc retrieval, query estimation by Dirichlet mixtures generally does not perform well, but aspects of its behavior show promise. Experiments where the original query is mixed with the models estimated by relevance models and Dirichlet mixtures confirms that query estimation methods should not fully discount the prior information held in a query.

1 Introduction

In the language modeling approach to information retrieval (IR), documents and queries are represented as probabilistic models [16]. Documents and queries are modeled as bags of words and their probabilistic equivalent is the multinomial model. The multinomial is easily understood as a biased die with words on its many faces. Typically documents are ranked by their likelihood of generating the query, i.e. query likelihood.

In bioinformatics, a set of genetic sequences with similar function define a sequence family. A probabilistic model of the family can be built and used to find other sequences that may likely be members of this family, or alternatively new sequences can be tested against all family models to find likely matches. When building a model to represent a family, the sequences are aligned and this creates a multiple alignment. Each sequence forms a row in the multiple alignment and each column is separately modeled. Each column of the alignment can be modeled with a multinomial model. The sequences in a family are a sample of some population of sequences that define the family. To enhance the generalizability of the family model, each column's multinomial model is smoothed.

Most generative models need to be smoothed to avoid zero probabilities. If a generative model has a zero probability of producing an element in a sample, it will be given a zero probability of producing the sample.

In IR, the document models are smoothed before measuring their likelihood of generating the query. A document with a zero probability for a term in the query means it would be impossible for the document to have generated the query. Commonly, the document's maximum likelihood model is linearly combined with the collection's maximum likelihood model so that a document has some probability of producing every word in the collection vocabulary.

Smoothing's goal goes beyond avoiding zero probabilities. Smoothing aims to produce better probability estimates for all words.

The true probabilistic model that generated a piece of text such as a document or query is unknown. In most cases, the text is a small sample. The proportions of the words present in the text are not the same as the probabilities of those words in the model that generated the text. In the extreme case, a word with non-zero probability in the model is not present in the sample – the zero probability problem.

While both documents and queries require accurately estimated models, considerably more research has focused on improving queries as opposed to documents. With queries being short and documents being relatively long, larger performance gains are likely to be had from improving the query model than from improving the document model.

This paper focuses on automatic methods to estimate query models without user interaction. Automatic query expansion and feedback methods are the more commonly studied techniques.

Automatic query expansion selects words to add to the query automatically without user supervision. Common methods involve adding words that are found to frequently co-occur with the query words.

Feedback methods take their inspiration from a user interaction technique called relevance feedback. After issuing a query and receiving a set of results from an IR system, a searcher can judge the relevance of a sample of the results. The IR system can use the relevant documents as large samples of text to better estimate the query model. A successful automatic variant of relevance feedback is local feedback, which is also known as pseudo or blind feedback. Local feedback *assumes* that the top k retrieved documents are relevant and performs relevance feedback before showing the results to the searcher. Feedback can be considered a form of query expansion, for feedback produces query models with many more non-zero probability words.

While never called smoothing methods, both automatic query expansion and local feedback are effectively smoothing techniques. These methods are attempting to better estimate their model of the user's query.

Sjölander et al. developed a sophisticated method, *Dirichlet mixtures*, for smoothing multinomial models in bioinformatics [17]. Given a sample, Dirichlet mixtures estimate the sample's true model using prior knowledge. In the context of IR, prior knowledge comes from the frequency of words and what words are used with other words.

Lavrenko and Croft's relevance models (RM) framework successfully produces better estimated query models through a local feedback-like process [13]. In this paper, we will show that relevance models can be seen as a special case of Dirichlet mixtures. We also discuss Dirichlet mixtures' relationship to probabilistic latent semantic indexing (pLSI) [5] and latent Dirichlet allocation [2].

To better understand how Dirichlet mixtures perform in the text domain, we apply them to the task of query estimation in a manner similar to relevance models.

The usual formulation of relevance models estimates a new model for the query and throws away the original model. Common to many query expansion techniques is the combination of the original query with the expanded model. Allan and Carterette have found that mixing the original query model with the RM estimated model produces better results than using only the RM model [1]. Zhai and Lafferty used a similar mixing in their model based feedback work [18]. Dirichlet mixtures provide insight into why this technique works and experiments are presented investigating the degree to which the original query model should be retained.

2 Methods and Materials

2.1 Text Modeling and Retrieval

A multinomial model of text specifies a probability for each word in the vocabulary V. The probabilities of the multinomial are its parameters and thus there are $|V|$ parameters, where $|V|$ is the number of words in the vocabulary. The probabilities of the multinomial sum to 1. A common way to think about the multinomial is as a biased die. A die has $|V|$ faces with each word having some probability of being *generated* by the die on a roll.

For a given piece of text, T, the parameters of the multinomial, M_T, representing T need to determined. This process of computing the probability of a word w given the model M_T, $P(w|M_T)$, is called estimation. A standard approach to parameter estimation is maximum likelihood estimation (MLE). MLE maximizes the likelihood of the observed data given the model. Treating the words of T as independent samples, the likelihood of T is defined to be:

$$L(T) = \prod_{w \in T} P(w|M_T)^{T(w)} \tag{1}$$

where $T(w)$ is the count of word w in T. The maximum likelihood estimate for the probability of a word turns out to be the count of that word divided by the total number of occurrences in T:

$$P(w|M_T) = \frac{T(w)}{|T|} \tag{2}$$

where $|T|$ is also known as the length of T and is defined as:

$$|T| = \sum_{w \in V} T(w) \tag{3}$$

The MLE model has zero probabilities for all words not in the sample of text. This is a problem for document retrieval.

2.2 Document Retrieval

Documents are ranked by how similar they are to the query. Given a document model M_D and a query model M_Q, the cross entropy measures how how well the document model encodes the query model:

$$H(M_Q|M_D) = -\sum_{w \in V} P(w|M_Q) \log P(w|M_D) \tag{4}$$

In this sum, $0 \log 0 = 0$. If for some word w, $P(w|M_D) = 0$ and $P(w|M_Q) > 0$, then the document will be given a score of $-\infty$. When the query model is the MLE model, cross entropy ranks equivalently to query likelihood:

$$P(Q|M_D) = \prod_{w \in Q} P(w|M_D)^{Q(w)} \tag{5}$$

The zero probabilities must be eliminated from the document models.

It is less clear that anything must be done to the query model, but queries are very small text samples. The MLE model of a query is inherently a poor representation of the true model that generated it. Both documents and queries should be smoothed to better estimate their text models.

2.3 Dirichlet Prior Smoothing

A solution to the problem of zero probabilities and poor probability estimates is to bring prior knowledge to the estimation process. A natural fit as a prior for the multinomial is the Dirichlet density [17]. A Dirichlet density can be thought of as a urn containing multinomial dies. All the multinomials are of the same size. In this paper, all multinomials have $|V|$ parameters. The Dirichlet density has the same number of parameters as the multinomials for which it is a prior. The vector $\vec{\alpha}$ represents the parameters of the Dirichlet density. For each word w in the vocabulary, there is a corresponding element α_w of $\vec{\alpha}$, and all $\alpha_w > 0$.

The estimate of the probability of word given a text, is now the weighted average of the word's probability in all multinomials. Each multinomial is weighted by its probability given the observed text and the Dirichlet density. This estimate is the mean posterior estimate:

$$P(w|M_T) = \int_M P(w|M) P(M|\vec{\alpha}, T) dM \tag{6}$$

which reduces to:

$$P(w|M_T) = \frac{T(w) + \alpha_w}{|T| + |\vec{\alpha}|} \tag{7}$$

as shown in [17]. The larger a sample of text, the less influence the prior has in determining the parameter estimates for the multinomial M_T. The mean of the Dirichlet density is for each α_w, $\alpha_w / |\vec{\alpha}|$. The shorter the text, the parameter estimates for M_T regress to the mean of the Dirichlet density. The bioinformatics community's common name for Equation 7 is *pseudo-counts*.

The parameters of the Dirichlet density can be determined using maximum likelihood estimation. MLE finds the density parameters that produce the highest likelihood for a collection of text samples when the density is used as a prior. The MLE can be computed numerically using a Newton-Raphson method [15] or via an expectation maximization (EM) like method [17].

The parameters of a Dirichlet density can be represented as a multinomial probability distribution M and a weight $m = |\vec{\alpha}|$. Thus, with $P(w|M) = \alpha_w / |\vec{\alpha}|$, Equation 7 becomes:

$$P(w|M_T) = \frac{T(w) + mP(w|M)}{|T| + m} \tag{8}$$

The machine learning community terms this formulation of Dirichlet prior smoothing the *m-estimate* [14]. The parameter m is the *equivalent sample size*. The Dirichlet density when used as a prior for the multinomial can be understood as taking m samples according to $P(w|M)$ prior to observing the data in T.

Dirichlet prior smoothing is a form of linear interpolated smoothing. Linear interpolated smoothing linearly combines two models to produce a smoothed model. As mentioned in the introduction, documents are typically smoothed with the collection. The document D is smoothed with the collection C as follows:

$$P(w|M_D) = (1 - \lambda) P(w|D) + \lambda P(w|C) \tag{9}$$

The λ parameter is varied between 0 and 1 to control the amount of smoothing.

Equation 8 can be written in the form of equation 9 [7] by setting λ in Equation 9 as follows:

$$\lambda = 1 - \frac{|T|}{|T| + m} \tag{10}$$

Thus Dirichlet prior smoothing can be seen as the mixing of two multinomial models. The amount of mixing depends on the length of text (sample size) relative to Dirichlet prior's equivalent sample size m. Common practice in IR is to use the collection model and empirically select m.

Dirichlet prior smoothing works well for smoothing documents and eliminating zero probabilities, but something more sophisticated should be used for smoothing a query.

2.4 Dirichlet Mixtures

Other than paying attention to the size of the text sample being smoothed, Dirichlet prior smoothing ignores the sample. Given a set of words, some words are more likely than others. Different topics though have different frequencies of word usage. Dirichlet mixtures estimate a set of priors. Each prior describes a different aspect or facet of how words are used with each other. The most likely priors given a sample have the most influence on determining the estimates of the smoothed model.

Rather than use only one Dirichlet density to provide prior information, a mixture of Dirichlet densities is used as the prior:

$$\rho = q_1\rho_1 + \ldots + q_n\rho_n \tag{11}$$

where ρ_i are the individual Dirichlet densities. All q_i are greater than zero and sum to 1. The q_i are known as the mixture coefficients. Each ρ_i has its own parameters $\vec{\alpha}_i$.

As shown in [17], the mean posterior estimate for a word w in a text T is now:

$$P(w|M_T) = \sum_{i=1}^{n} P(\vec{\alpha}_i|T,\Theta)\frac{T(w)+\alpha_{i,w}}{|T|+|\vec{\alpha}_i|} \tag{12}$$

where Θ represents all the parameters of the Dirichlet mixture. Dirichlet mixtures weight the prior information of each density by the probability of the density given the text and mixture. Dirichlet mixtures allows different densities to exert different prior weight with varying equivalent sample sizes. Some densities can be very general with low prior weight while others are very specific and should have much more influence over a matching query. Thus, Dirichlet mixtures could contain prior knowledge ranging from term frequencies in the entire collection to broad topics, individual documents or even passages.

Dirichlet mixtures, like the single density Dirichlet prior smoothing, determine the degree to which the original sample is retained based on its size. The larger the sample, the more influence the query retains.

The parameters of a mixture of Dirichlet densities are determined using an expectation maximization (EM) like process to maximize the likelihood of a set text samples when smoothed using the mixture [17]. The number of densities n is selected manually.

Similar to how the single density Dirichlet prior smoothing can be written as linear interpolated smoothing, Dirichlet mixtures can be rewritten as:

$$\begin{aligned} P(w|M_T) = P(w|T)\sum_{i=1}^{n} P(\vec{\alpha}_i|T,\Theta)(1-\lambda_i) + \\ \sum_{i=1}^{n} \lambda_i P(\vec{\alpha}_i|T,\Theta) m_i P(w|M_i) \end{aligned} \tag{13}$$

where

$$\begin{aligned} \lambda_i &= 1 - \frac{|T|}{|T|+|\vec{\alpha}_i|} \\ P(w|T) &= T(w)/|T| \\ m_i &= |\vec{\alpha}_i| \\ P(w|M_i) &= \alpha_{i,w}/|\vec{\alpha}_i| \end{aligned}$$

With this formulation, one can see that the original text sample T is weighted by some fixed amount that reflects the average degree of smoothing each density in the mixture performs. The sample is smoothed with a set of multinomials whose influence is governed by their equivalent sample size m_i and most importantly by how likely the density is given the sample, which includes the density's prior likelihood q_i.

The spirit of Dirichlet mixtures is to smooth a text sample by finding a set of models and mixing them with the text sample in proportion to their similarity with the sample:

$$\begin{aligned} P(w|M_T) = (1-\lambda)P(w|T) + \\ \lambda \sum_{M} P(M_i|T)P(w|M_i) \end{aligned} \tag{14}$$

An entire family of smoothing methods could be developed and studied by determining:

1. λ: How much to discount the original sample.
2. M: The set of models.
3. $P(M_i|T)$: How to weight each model.

We next describe three techniques that can be seen as fitting within this smoothing family. Each of relevance models, probabilistic latent semantic indexing, and latent Dirichlet allocation completely discount the original text sample, i.e. $\lambda = 1$.

2.5 Relevance Models

In Lavrenko and Croft's relevance models (RM) framework, a model is considered to have generated both the query and the document and this model is called a relevance model [13]. Once the relevance model is calculated, it replaces the query and cross entropy is used to rank the documents. As used for ad-hoc retrieval, relevance models is a local feedback technique.

The relevance model M_R is calculated as the weighted average of the all documents D in the collection:

$$P(w|M_R) = \sum_D P(D|Q)P(w|D) \tag{15}$$

where Q is the query entered by the user. Each document model is weighted by its probability given the query. Using Bayes' rule, $P(D|Q)$ can be calculated as follows:

$$\begin{aligned} P(D|Q) &= \frac{P(D)P(Q|D)}{P(Q)} & (16)\\ &= \frac{P(D)\prod_{w\in Q} P(w|D)^{Q(w)}}{\sum_D P(D)\prod_{w\in Q} P(w|D)^{Q(w)}} & (17)\end{aligned}$$

Lavrenko and Croft demonstrate and explain that the equation for $P(D|Q)$ is dominated by the $\prod_{w\in Q} P(w|D)^{Q(w)}$ term in the numerator, which is also the query likelihood measure used for document retrieval. The query likelihood values rapidly decrease with increased ranks (rank 1 is the best document). Thus equation 15 can be approximated by using only the top k documents returned from an initial document retrieval using the query Q. Often k is set to 50. In addition, a uniform prior probability is assumed for documents and thus $P(D)$ can be dropped from equation 16. Equation 15 becomes:

$$P(w|M_R) = \sum_{i=1}^{k} P(D_i|Q)P(w|D_i) \tag{18}$$

where $P(D_i|Q)$ is calculated by:

$$P(D_i|Q) = \frac{P(Q|D_i)}{\sum_{j=1}^{k} P(Q|D_j)} \tag{19}$$

As mentioned in the introduction, the original query model is often mixed with the relevance model to help keep the query "focused":

$$P(w|M_Q) = (1-\lambda)P(w|Q) + \lambda P(w|M_R) \tag{20}$$

which is a linear interpolated smoothing of the query with the relevance model. Equation 20 is a special case of Dirichlet mixtures. This special case can be written as:

$$P(w|M_Q) = \sum_{i=1}^{k} P(D_i|Q)\frac{Q(w) + mP(w|D_i)}{|Q|+m} \tag{21}$$

The text sample T of equation 12 is now the query Q. The Dirichlet densities $\vec{\alpha}_i$ are represented as multinomials D_i and all have an equal equivalent sample size m. In the same way that relevance models assume a uniform probability for all documents, now all q_i are uniform and can be ignored. To get from equation 21 to 20 requires only a few algebraic manipulations. First, using equation 10, the right hand side of equation 21 can be rewritten as:

$$\sum_{i=1}^{k} P(D_i|Q)((1-\lambda)P(w|Q) + \lambda P(w|D_i)) \tag{22}$$

where $\lambda = 1 - |Q|/(|Q|+m)$. Further manipulation produces:

$$(1-\lambda)P(w|Q)\sum_{i=1}^{k} P(D_i|Q) + \lambda\sum_{i=1}^{k} P(D_i|Q)P(w|D_i)$$

The sum $\sum_{i=1}^{k} P(D_i|Q)$ is equal to 1 given equation 19 and thus we finally get:

$$P(w|M_Q) = (1-\lambda)P(w|Q) + \lambda\sum_{i=1}^{k} P(D_i|Q)P(w|D_i)$$

which is the same as equation 20 if the k documents chosen are the top k from an initial retrieval.

In terms of equation 14, RM completely discounts the original query, uses the documents in a collection as the models, and weights these models by their probability given the query.

2.6 pLSI

Hofmann's probabilistic latent semantic indexing (pLSI) is a technique for representing text in terms of a set of learned topics Z [5]. For a given piece of text T, the probability of a word w is given by:

$$P(w|M_T) = \sum_Z P(Z_i|T)P(w|Z_i) \tag{23}$$

Each topic Z_i is a multinomial. The topics and their weights are found with an expectation maximization (EM) process. While Dirichlet mixtures directly calculates the probability of a density given a sample, $P(\vec{\alpha}_i|T,\Theta)$, pLSI empirically finds these weights for each document as it also searches the space of topics. The likelihood of the documents is maximized given a preselected number of topics.

The aim is for the topics to capture connections between words that are not possible from co-occurrences at the document or passage level. For example, British documents about cars are likely to mention petrol while US-centric documents would mention gas. The words petrol and gas do not co-occur with high frequency, but the words surrounding them can tell us that gas and petrol are similar words. These are the sorts of connections between words that are considered the latent semantic information present in the collection. Sometimes the topics are referred to as *aspects* or *facets*.

Given a new sample of text, the same EM process is used to find the $P(Z_i|T)$ weights but with the set of topics fixed. Hofmann calls this "folding in" the text sample. Aspect based models such as pLSI can represent text samples in a compressed form by only storing the computed $P(Z_i|T)$ weights. Equation 23 "uncompresses" the text giving us the model M_T. This is a lossy form of compression, i.e. the original cannot be recovered.

2.7 Latent Dirichlet Allocation

Latent Dirichlet allocation (LDA) is another model similar to Dirichlet mixtures [2]. LDA is a mixture of multinomial models. These models are similar to the topics in pLSI. The models represent the "latent" aspects of a collection. A Dirichlet distribution determines the weight of each model in the mixture. For a sample of text, its weighting Dirichlet distribution is found using a process similar to the folding in process used by pLSI [4]. Lavrenko explains in great detail the connections between relevance models, pLSI, and LDA [11].

2.8 Prior Value of Query

Each of RM, pLSI, and LDA can be seen as fitting in the family of smoothing methods inspired by Dirichlet mixtures described by equation 14. On the other hand, only Dirichlet mixtures is explicitly a prior on the multinomial model. These other models can take a text sample and expand it with similar terms. When the expanded model is linearly mixed with the original text sample, each becomes a smoothing method.

Retaining the influence of the query has been a part of feedback techniques since at least that of Rocchio's relevance feedback work [6]. Rocchio chose to retain the original query for relevance feedback because the user feedback was known to only be a sample of the documents.

When local feedback is seen as smoothing of the query, Dirichlet mixtures offers an another explanation for retaining the query. Dirichlet mixtures explicitly models the prior value of the sample. The larger the sample is relative to a single Dirichlet density's equivalent sample size m, the more prior weight is given to the counts in the sample.

We investigate the extent to which the original query should be retained when mixed with the model computed by RM as per equation 20.

3 Experiments

Two experiments were conducted to better understand the behavior of Dirichlet mixtures (DM) when applied to smoothing of probabilistic models of text. The first experiment was to use Dirichlet mixtures in a manner similar to relevance models for query estimation using local feedback. The second experiment examined the effect of re-mixing the models produced by both methods with the original query model. The mixing is the linear interpolation of the MLE query model with the model produced by the method. Equation 20 shows the mixing for relevance models.

Given the stochastic process for estimating the Dirichlet mixtures parameters, several runs should be conducted and averaged. Because of the computational expense, only one run per parameter settings was conducted.

Query likelihood with Dirichlet prior smoothing formed the baseline retrieval. We performed of sweep of the Dirichlet prior parameter m to determine a reasonable setting. The following values were tried for m: {50, 100, 150, 200, 250, 300, 350, 400, 500, 600, 800, 1000, 1250, 1500, 1750, 2000, 2500, 3000, 5000}. Setting m to 800 produced the highest mean average precision (MAP) of 0.217 on title queries for topics 351-450. On description queries, $m = 3000$ produced the highest MAP of 0.205. Dirichlet prior smoothing is robust to settings of m so long as it is not set too small [19]. A compromise setting of $m = 1500$ was used for all experiments. At this setting of m, the title queries have a MAP of 0.214, and description queries have a MAP of 0.203. The larger setting of m was also chosen because the expanded queries produced by Dirichlet mixtures (DM) and relevance models are likely to be more similar to the longer description queries and thus would benefit from a larger m.

The baseline was used to determine the top k documents for blind feedback used by both RM and DM. On title and description queries, k was tried at values of 20, 50, and 100 documents. The number of densities (NOD) in the Dirichlet mixture was set for k=20, NOD = {2,4}, for k=50, NOD = {5,10}, and for k=100, NOD={5,10,20}. This produces an "average size" of components of 5, 10, and 20.

Dirichlet mixtures were estimated using 500 iterations of the process detailed in [17]. We initialized the densities by setting a term's Dirichlet parameter, α_w, to be the term's frequency in the top k documents divided by the number of documents. We then add a small amount of noise to each density. The effect is to start with the densities near each other and near the "center" of the space.

The expanded models produced by both RM and DM were truncated to the 50 highest probability terms. We tested the performance of RM with the model truncated at both 50 and 1000 words and tested with $k = \{20, 100\}$ on both title and description queries. Table 1 shows that the smaller models have the same mean average precision, slightly lower precision at 20 documents, but a significantly higher number of relevant documents retrieved. To favor higher recall, 50 word models were used for all RM and DM experiments.

Relevance models (RM) were run with the the Lemur feedback coefficient set to 0.6, which matches the smoothing between documents and collection reported in [12]. Relevance models version 1, RM1, was used in all experiments.

To test our Dirichlet mixtures code, we obtained Kevin Karplus' code to generate (amino acid) sequences from a Dirichlet mixture [9]. After getting the code running, we had it generate 1000 sequences of the default mixture. We treated each sequence as a document and each of the 20 amino acids as a term and indexed this as a mini document collection. We then had our code estimate a Dirichlet mixture for the 1000 sequences. We gave the code the correct number of components. After 3000 iterations, the estimated mixture was within 1-5% of the correct mixture for nearly all of the parameters. A couple parameters were off by up to 20%. Thus, our code tested as correct.

For the second experiments, a sweep of λ values was performed to determine the best weight for the original MLE query model and the truncated models produced by Dirichlet mixtures and relevance models. The parameter λ controls the amount of linear mixing as per equation 20. Values of λ included $\{0.01, 0.02, \ldots, 0.09, 0.1, 0.2, \ldots, 0.9\}$. An initial sweep between 0.1 and 0.9 found that some runs had a maximum mean average precision at 0.1 and so a detailed sweep was performed between 0.01 and 0.1.

Statistical significance is measured using a two-sided, paired, randomization test with 10000 samples (see page 168 [3]). Unless otherwise stated, significance is at the $p < 0.05$ level.

3.1 Topics and Collection

The topics used for the experiments consists of TREC topics 351-450, which are the ad-hoc topics for TREC 7 and 8. TREC topics consist of a short title, a sentence length description, and a paragraph sized narrative. The titles best approximate a short keyword query. The descriptions are more representative of a verbose query, which is more likely to contain common non-informative words. The experiments use the titles and descriptions separately.

The collection for the TREC 7 and 8 topics consists of TREC volumes 4 and 5 minus the CR subcollection. This 1.85 GB, heterogeneous collection contains 528,155 documents from the Financial Times Limited (FT), the Federal Register (FR), the Foreign Broadcast Information Service (FBIS), and the Los Angeles Times (LAT).

We preprocessed the collections and queries in the same manner. The Krovetz stemmer [10] stemmed all words. Words from an in-house stop word list of 418 noise words were removed. 649,929 unique terms comprise the resulting vocabulary. The average document length is 270 words. We used Lemur 3.1 [20] for all experiments.

4 Results

Table 2 shows the performance of Dirichlet mixtures vs. relevance models for query estimation. The baseline is the query likelihood run, which is the run that produces the initial retrieval that both local feedback methods use to re-estimate the query.

Table 3 shows the results of taking the re-estimated models produced by Dirichlet mixtures and relevance models and mixing their estimated models with the MLE query model.

5 Discussion

Table 2 shows that only the Dirichlet mixtures run on title queries with $k = 20$ and with 2 densities performed near relevance models (RM). On the title queries, with $k = 20$, RM achieved a mean average precision (MAP) of 0.239. In comparison, the Dirichlet mixtures ($k = 20$, #Dens=2) run achieved a MAP of 0.231. The difference between the two runs is not statistically significant, but across all measures RM appears better than Dirichlet mixtures.

Lavrenko has reported results for RM on five other collections and sets of title queries, and on these RM has a percentage improvement in MAP of between 15.6% and 26.5% [11]. The best performance for RM

Query	*k* Docs	Model Size	MAP	P20	Num. Rel. Ret.
Title	20	50	0.239	0.364	5808
Title	20	1000	0.242	0.387	5507
Title	100	50	0.243	0.361	5784
Title	100	1000	0.243	0.381	5426
Desc.	20	50	0.213	0.331	5152
Desc.	20	1000	0.214	0.357	4883
Desc.	100	50	0.216	0.337	5234
Desc.	100	1000	0.214	0.353	4915

Table 1: This table shows the effect of model truncation on the relevance model. Using the 50 highest probability words produces an equivalent mean average precision (MAP) as using 1000 words. The smaller 50 word models show less precision at 20 documents (P20) but significantly higher number of relevant documents retrieved. One thousand documents were retrieved for each topic.

Query	Method	k	#Dens	MAP	Pct.	P20	Pct.	Recall	Pct.
Title	QL			0.214		0.368		0.526	
Title	RM	20		0.239	11.4	0.364	-1.1	0.618	17.4
Title	RM	50		0.241	12.5	0.360	-2.0	0.619	17.6
Title	RM	100		0.243	13.3	0.361	-1.9	0.615	16.9
Title	DirMix	20	2	0.231	7.6	0.361	-1.9	0.603	14.6
Title	DirMix	20	4	0.217	1.1	0.354	-3.7	0.567	7.8
Title	DirMix	50	5	0.208	-3.0	0.318	-13.5	0.573	9.0
Title	DirMix	50	10	0.204	-4.9	0.322	-12.5	0.553	5.2
Title	DirMix	100	5	0.199	-7.3	0.298	-18.9	0.562	6.8
Title	DirMix	100	10	0.204	-4.9	0.331	-10.1	0.583	10.9
Title	DirMix	100	20	0.207	-3.2	0.327	-11.0	0.545	3.6

Query	Method	k	#Dens	MAP	Pct.	P20	Pct.	Recall	Pct.
Desc.	QL			0.203		0.341		0.515	
Desc.	RM	20		0.213	5.1	0.331	-2.9	0.548	6.3
Desc.	RM	50		0.215	5.9	0.333	-2.3	0.553	7.3
Desc.	RM	100		0.216	6.6	0.337	-1.2	0.557	8.0
Desc.	DirMix	20	2	0.191	-6.1	0.309	-9.4	0.542	5.2
Desc.	DirMix	20	4	0.164	-18.9	0.279	-18.3	0.517	0.4
Desc.	DirMix	50	5	0.144	-29.0	0.245	-28.3	0.474	-8.0
Desc.	DirMix	50	10	0.136	-32.8	0.234	-31.4	0.426	-17.3
Desc.	DirMix	100	5	0.144	-29.2	0.248	-27.3	0.486	-5.7
Desc.	DirMix	100	10	0.137	-32.3	0.238	-30.4	0.474	-8.1
Desc.	DirMix	100	20	0.117	-42.3	0.215	-37.0	0.416	-19.3

Table 2: This table shows the performance of Dirichlet mixtures and relevance models compared to the baseline retrieval for TREC topics 351-450. The baseline, QL, is the query likelihood run that provides the top *k* documents that each local feedback method uses to re-estimate the query. The measures shown are the mean average precision (MAP), precision at 20 documents retrieved (P20), and recall at 1000 documents. For each measure, the percent change relative to the baseline is also shown (Pct.). Blanks are “not applicable.”

Query	Method	k	#Dens	Query Wgt	Model Wgt	Orig. MAP	New MAP	Pct.
Title	QL					0.214	0.214	
Title	RM	20		0.03	0.97	0.239	0.256	19.5
Title	RM	50		0.03	0.97	0.241	0.257	19.8
Title	RM	100		0.02	0.98	0.243	0.257	19.8
Title	DirMix	20	2	0.06	0.94	0.231	0.250	16.8
Title	DirMix	20	4	0.07	0.93	0.217	0.247	15.4
Title	DirMix	50	5	0.10	0.90	0.208	0.234	9.2
Title	DirMix	50	10	0.10	0.90	0.204	0.234	9.1
Title	DirMix	100	5	0.10	0.90	0.199	0.228	6.3
Title	DirMix	100	10	0.10	0.90	0.204	0.232	8.5
Title	DirMix	100	20	0.09	0.91	0.207	0.238	10.9

Query	Method	k	#Dens	Query Wgt	Model Wgt	Orig. MAP	New MAP	Pct.
Desc.	QL					0.203	0.203	
Desc.	RM	20		0.09	0.91	0.213	0.240	18.4
Desc.	RM	50		0.08	0.92	0.215	0.239	18.0
Desc.	RM	100		0.08	0.92	0.216	0.239	17.9
Desc.	DirMix	20	2	0.20	0.80	0.191	0.223	9.8
Desc.	DirMix	20	4	0.20	0.80	0.164	0.218	7.3
Desc.	DirMix	50	5	0.40	0.60	0.144	0.207	2.2
Desc.	DirMix	50	10	0.40	0.60	0.136	0.209	3.2
Desc.	DirMix	100	5	0.30	0.70	0.144	0.211	4.0
Desc.	DirMix	100	10	0.40	0.60	0.137	0.208	2.8
Desc.	DirMix	100	20	0.50	0.50	0.117	0.207	1.8

Table 3: This table shows the results of taking the re-estimated models produced by Dirichlet mixtures and relevance models and mixing their estimated models with the MLE query model. Results are shown for TREC topics 351-450. The baseline is the query likelihood (QL) run that provides the top k documents to each local feedback method. The weight given to the original query model is in the "Query Wgt" column, while the corresponding weight of the expanded models is in the "Model Wgt" column. Listed is the mean average precision (MAP) for the methods without the re-mixing with the query and also the new MAP that results from re-mixing with the query. The percent change (Pct.) of the new MAP compared to the baseline is also shown. Blanks are "not applicable."

on topics 351-450 in table 2 was a 13.3% improvement in MAP. Topics 351-450 and their associated collection can thus be considered to be relatively hard for the task of query estimation. A possible reason for the difficulty may come from the heterogeneous collection. RM results have previously been reported on collections with documents all from the same source.

Offering more documents for Dirichlet mixtures and more densities in its mixture leads to performance degradation. All methods have difficulty with the description queries compared to the title queries. Dirichlet mixtures though fail on the description queries with decreases in MAP of -6.1% to -42.3% compared to the baseline.

5.1 Observations

It's an open issue on how best to initialize the parameters of a mixture. The initial conditions for the hill climbing are not specified in [17]. Karplus describes his experience with learning a Dirichlet mixtures model [8]. As with EM, each search can result in a different set of model parameters. The estimation process contains a learning rate η. If η is set too high, the model explodes or the parameter values oscillate wildly. If set too low, little progress is made and the process won't converge. The learning rate needs to change based on the number of densities and the number of documents. The more documents, the slower the learning rate needs to be. The more densities, the faster the rate needs to be.

Hofmann shows that if pLSI learns the wrong facets of a dataset, then its ability to model the data degrades quickly [5]. Hofmann had problems with the facets congealing too quickly and incorrectly. To solve this problem, he used tempered EM, which slows down the rate at which EM converges.

Unlike our test of learning the parameters for the amino acid sequences, the documents quickly cluster and don't seem to straddle densities. This hard clustering of documents typically occurs within 10 iterations of the estimation process. Clusters vary in size. While a mixture may be allowed 10 or 20 densities, typically a fewer number of clusters forms and a number of the densities are garbage with no weight given to them. This may point to the issue of having a larger vocabulary than number of documents, which may allow for overfitting. In the case of learning the parameters for genetic sequences, the number of sequences is much greater than the size of the vocabulary. Additionally, clusters appear to prefer to form around the various subcollections. This may be because similar sources use similar language, but it could also be because similar sources have similar junk in them. Junk markup makes little difference for retrieval, since users don't enter anything that would match the junk. For clustering of documents though, the junk could be a stronger attractor than the documents' content words.

As the estimation process determines the correct $\vec{\alpha}$ values for each density, $|\vec{\alpha}|$ tends to at first grow and then shrink. If the process is stopped too soon, the densities' equivalent size ($m = |\vec{\alpha}|$), will be too large.

5.2 Query Analysis

Dirichlet mixtures chief problem appears to be that the densities it learns are fit very closely to the documents that cluster under them. A density can effectively have zero probabilities for terms in the query. Densities with zero probabilities will have a zero $P(\vec{\alpha}_i|T, \Theta)$ in equation 12.

For example, the Dirichlet mixtures run with $k = 20$ and four densities was *almost* successful on description topic 441. This topic is "prevent treat lyme disease." The mixture found the lyme disease documents among the top 20 documents. Almost all documents discussing lyme disease cluster under one density. This density had a tiny probability for "prevent" – effectively a zero probability. Another density had non-zero probabilities for all four words. This other density was primarily about infectious diseases such as AIDS and had a high probability for "prevent." The estimated query model thus focused on diseases and their prevention with little weight given to the issue of lyme disease.

On title query 418, "quilts, income," the Dirichlet mixtures run with $k = 2$ and 2 densities fails. A small cluster is formed with four documents about retirement, income and the AIDS quilt. A large set of documents about quilts forms the other density. The quilts oriented density lacks the term income. The final query model is built from the wrong density and average precision is 0. A sparse space can cluster strangely leading to disaster.

As the number of densities increases, the chance of zero probabilities increases. This doesn't explain though why with $k = 50$ and 5 densities, Dirichlet mixtures performed worse than with $k = 20$ and 2 densities. Each of these runs should have had an average of 10 documents per density. Providing Dirichlet mixtures with more documents should help it. If the densities being formed are more often than not garbage, then by restricting the number feedback documents, the damage is limited. Higher ranked documents are more likely to relevant.

On title query 372, the Dirichlet mixtures run with $k = 100$ and 10 densities fails for a confusing reason.

The query is "Native American casino" and the mixture forms a density around the documents discussing casinos and gambling. For some unknown reason, the term "casino" is effectively zero in this density. The term appears frequently in the documents clustered under the density. We tested our code to make sure it could correctly estimate a mixture of Dirichlet densities, but the estimation process is complex and stochastic. This complexity makes it difficult to obtain an efficient, and apparently perfect implementation.

Dirichlet mixtures do show promise. On some queries the technique performs significantly better than relevance models. On the title query 405, "cosmic events," the searcher is looking for recent astronomical phenomena and not necessarily information about cosmic rays. On this topic, Dirichlet mixtures with $k = 20$ and 2 densities has an average precision of 0.285. Relevance models with $k = 20$ has an average precision of 0.087. Even when the original query is mixed into the models produced by these methods, Dirichlet mixtures significantly outperforms relevance models.

For topic 405, the top 20 documents contain science and non-science articles. Dirichlet mixtures forms two densities around these areas. The computed probability of a density given the query is 0.8 for the science density and 0.2 for the other. The documents were nearly evenly split between the two densities with 55% in the science density. The science density has a higher probability for both terms.

Relevance models does not fail on this query because it fails to find the science articles. It fails because it focuses on the top ranked documents which stress cosmic rays and solar radiation. Dirichlet mixtures produces a model that is more general and emphasizes the word "universe." Many relevant documents are of the sort "scientists have discovered the biggest thing in the universe."

In this case, the more general collection of documents equally weighted in the science density outperformed the focused top documents used by relevance models. All of the documents in the density contribute equally to its makeup regardless of their individual distance to the original query. As always, Dirichlet mixtures could have simply gotten lucky, but this is an interesting case where it succeeded.

On title query 396, "sick building syndrome" the correct densities are again found by Dirichlet mixtures with $k = 20$ and 2 densities. One density focuses on sick building syndrome and the other has a collection of documents mentioning buildings or other syndromes. The clustering behavior has the potential of cleaning up the documents. The correct density ends up having the most influence on the smoothing the query. Again, this success could be random luck, but it is the behavior one hopes of Dirichlet mixtures.

5.3 Re-mixing with Query

As has been previously discovered [1, 18], mixing the original query with the expanded model can produce significant gains. On title queries, RM goes from a mean average precision (MAP) performance improvement over the baseline of 11.4%-13.3% without re-mixing to 19.5%-19.8% with re-mixing. On description queries, RM leaps from a MAP improvement of 5.1%-6.6% without re-mixing to 18.4%-17.9% with re-mixing.

While the Dirichlet mixture models have already incorporated the original query, the equivalent sample sizes for the densities often ranged in value from at least 300 to 1000 or greater. Relative to the size of the query, Dirichlet mixtures ignores the counts of query. The high equivalent sample sizes may be an artifact of the estimation process. If the estimation process does not have enough time to relax, the equivalent sample sizes will be artificially high. Another possible cause is that documents are used as the samples, which have an average length of 270 words. In the bioinformatics application of Dirichlet mixtures, the sample to be smoothed is approximately the same size as the samples used to estimate the model.

When looking at the Dirichlet mixture runs in table 3, the re-mixing improves the performance of Dirichlet mixtures. The best Dirichlet mixtures run goes from a MAP performance improvement of 7.6% to 16.8%. Re-mixing with the query resuscitates the performance of the title run with $k = 20$ and 4 densities from an improvement of 1.1% to 15.4%. The poorly performing Dirichlet mixture runs require significant weight added to the original query.

At first glance, table 3 validates Dirichlet mixtures behavior of giving more prior weight to longer queries. For RM runs with the short title queries, the best retrieval performance is had by giving the original MLE query model a weight around 0.03. For RM runs with the longer description queries, the query model does best with a weight around 0.08.

The average query length is 2.44 words for title queries and 7.84 words for description queries. In the MLE query models, the average probability given to a term is 0.41 for title and 0.13 for description queries. When multiplied by the weights 0.03 and 0.08, the probabilities in these models become 0.0123 and 0.0104. These probabilities are nearly the same. In addition, these probabilities are close to the top probabilities of the computed relevance models. Thus, rather than giving more weight to longer queries because they are better samples, it appears important in this collection to equally weight the contribution from the query and expanded model on a per term basis.

This matches the results reported by Zhai and Lafferty [18]. They created an expanded model which they then truncated and renormalized. While the optimal amount of mixing varied across text collections, they report that a nearly equal mixing of query and expanded model is the safe bet. The truncated models used in our experiments are not renormalized. If we had renormalized them, a λ near 0.5 is likely to have been the result for both title and description queries.

Nevertheless, treating the query as prior information that has some weight is an valuable way to look at the success found from mixing the query with expanded models.

6 Future Work

Dirichlet mixtures, pLSI, and LDA are all aspect-based techniques. Using computationally expensive methods, these methods take a collection of documents and represent it in a reduced form. Both pLSI and LDA use a set of multinomials while Dirichlet mixtures uses Dirichlet densities. This difference allows Dirichlet mixtures to have different amounts of influence over text samples given the size of the sample.

Aspect-based models attempt to directly model the connections found between words such as gas and petrol. In the set of densities in Dirichlet mixtures or the multinomials of pLSI and LDA, should be a density or multinomial that has high probability for both gas and petrol and their associated words. Assuming gas and petrol never co-occur in a collection of documents, relevance models cannot smooth the word petrol and produce a model that also gives the word gas high probability. This matters little for ad-hoc retrieval. The relevance model will pick up all the words co-occurring with petrol. When the relevance model is then used for retrieval, it will pull up all the documents mentioning gas.

Nevertheless, as the query analysis showed, there appears to be some value in mixing with models more general than individual documents. Dirichlet mixtures makes it clear that both coarse and fine grain priors can be used. One prior could be the entire collection while another is a passage. Because the text sample self selects the priors most likely for it, there should be little risk in expanding the set. The expensive estimation process currently used by Dirichlet mixtures could be replaced by a more affordable hierarchical clustering of a collection. This clustering would allow for the creation of models at different levels of the hierarchy. Equivalent sample sizes could be assigned to the models based on the number of documents covered by a model in the hierarchy.

The priors used in Dirichlet mixtures do not have to come only from the retrieval collection. A user's past search behavior or a user's personal document collection could be used as priors by which the query model should be smoothed. Such changes to the query would allow for personalized search.

7 Conclusion

We investigated the use of the Dirichlet mixtures smoothing technique in the text domain by applying the technique to the problem of query estimation. Dirichlet mixtures originated in bioinformatics for better estimation of the model of a column in a multiple alignment. Dirichlet mixtures have difficulty with the large vocabulary size of information retrieval. For this reason, Dirichlet mixtures perform below that of relevance models or the baseline on the task of query estimation. On some queries, Dirichlet mixtures perform very well and this shows that there may be value to utilizing aspect-based prior information. Inherent in Dirichlet mixtures is the incorporation of a prior value of the text sample, which gives insight into the success of re-mixing the query with the expanded model produced by relevance models. As utilized, Dirichlet mixtures failed to give enough weight to the original query and performed better with a re-mixing of the query. For relevance models, a near equal per term weighting of the original query model probabilities and the expanded model probabilities appears to be more important than the small average difference in sample size between title and description queries.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval and in part by SPAWARSYSCEN-SD grant number N66001-02-1-8903. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

References

[1] James Allan and Ben Carterette. Personal communication, 2004. James Allan originated the idea to mix the relevance model with the original query model. Ben Carterette performed preliminary experiments showing significant gains in performance are possible with this formulation.

[2] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January 2003.

[3] Paul R. Cohen. *Empirical methods for artificial intelligence.* MIT Press, 1995.

[4] Mark Girolami and Ata Kabán. On an equivalence between plsi and lda. In *SIGIR*, pages 433–434, 2003.

[5] Thomas Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, pages 50–57, 1999.

[6] Jr. J. J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART Retrieval System*, pages 313–323. Prentice Hall, 1971. Originally published as Section XXIII of Scientific Report ISR-9, August 1965.

[7] W. E. Johnson. Probability: deductive and inductive problems. *Mind*, 41(164):409–423, 1932.

[8] Kevin Karplus. Regularizers for estimating distributions of amino acids from small samples. Technical Report UCSC-CRL-95-11, Baskin Center for Computer Engineering and Information Sciences, UCSC, 1995.

[9] Kevin Karplus. Gen_sequence, 2000. `http://www.cse.ucsc.edu/research/compbio/gen_sequence/`.

[10] Robert Krovetz. Viewing morphology as an inference process. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 191–202. ACM Press, 1993.

[11] Victor Lavrenko. *A Generative Theory of Relevance.* PhD thesis, University of Massachusetts Amherst, 2004.

[12] Victor Lavrenko and W. Bruce Croft. Relevance based language models. In *SIGIR*, pages 120–127, 2001.

[13] Victor Lavrenko and W. Bruce Croft. Relevance models in information retrieval. In W. Bruce Croft and John Lafferty, editors, *Language Modeling for Information Retrieval*, volume 13 of *The Kluwer International Series on Information Retrieval.* Kluwer Academic Publishers, 2003.

[14] Tom M. Mitchell. *Machine Learning.* McGraw-Hill, 1997.

[15] A. Narayanan. Algorithm as 266: Maximum likelihood estimation of the parameters of the dirichlet distribution. *Applied Statistics*, 40(2):365–374, 1991.

[16] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *SIGIR 1998*, 1998.

[17] Kimmen Sjölander, Kevin Karplus, Michael Brown, Richard Hughey, Anders Krogh, I. Saira Mian, and David Haussler. Dirichlet mixtures: a method for improved detection of weak but significant protein sequence homology. *Comput. Appl. Biosci.*, 12:327–345, August 1996.

[18] Chengxiang Zhai and John Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 403–410. ACM Press, 2001.

[19] ChengXiang Zhai and John Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 334–342. ACM Press, 2001.

[20] ChengXiang Zhai, Thi Nhu Truong, John Lafferty, Jamie Callan, David Fisher, Fangfang Feng, James Allan, Bruce Croft, Paul Ogilvie, Bill Jerome, Adam Berger, Imre Kondor, and Victor Lavrenko. The lemur toolkit for language modeling and information retrieval. `http://www.cs.cmu.edu/~lemur/`.