Bayesian extension to the Language Model for Ad Hoc Information Retrieval

Hugo Zaragoza
Microsoft Research
Cambridge, U.K.
hugoz@microsoft.com

Djoerd Hiemstra
University of Twente
The Netherlands
hiemstra@cs.utwente.nl

Michael Tipping
Microsoft Research
Cambridge, U.K.
mtipping@microsoft.com

ABSTRACT

We propose a Bayesian extension to the ad-hoc Language Model. Many smoothed estimators used for the multinomial query model in ad-hoc Language Models (including Laplace and Bayes-smoothing) are approximations to the *Bayesian predictive distribution*. In this paper we derive the full predictive distribution in a form amenable to implementation by classical IR models, and then compare it to other currently used estimators. In our experiments the proposed model outperforms Bayes-smoothing, and its combination with linear interpolation smoothing outperforms all other estimators.

Categories and Subject Descriptors

H [**3**]: 3—*Retrieval models*

General Terms

Algorithms, Performance, Theory

Keywords

Information Retrieval, Ad Hoc Retrieval, Ad Hoc Language Model, Bayesian Language Model

1. INTRODUCTION

In the last five years, Language Models have become one of the most promising areas in which progress in information retrieval theory and practice is expected. A Language Model computes the relevance of a document d with respect to a query q by estimating a factorised form of the distribution $P(q, d)$ [9][8][5]. There have been several alternative formulations and derivations of the Language Model, as well as different generalisations [6][1]. Despite the fact that, in some ways, they remain controversial [10], Language Models are of great interest: they are elegant mathematical models for ad-hoc retrieval and have shown to produce excellent empirical results.

In this paper, we propose a Bayesian extension of the Language model. Bayesian statistics provide very useful concepts and tools for estimation which can be of great interest to the Information Retrieval community. They provide a powerful yet intuitive mathematical framework for data modelling when the data is scarce and/or uncertain and there is some prior knowledge about it. In particular we are interested in Bayesian statistics for the Language Model because it does away with the need to explicitly *smooth* parameters. Rather, the uncertainty of the data is taken into account in the inference framework.

In fact, one of the best smoothing techniques used today in Language Models is *Bayes-smoothing* or *Dirichlet smoothing* [14]. Bayes-smoothing is an approximation to the full Bayesian inference model: in fact, it is the *maximum posterior* approximation to the *predictive distribution*. In this paper we derive analytically the predictive distribution of the most commonly used query Language Model, and show experimentally that i) it improves performance over the usual Bayes-smoothing solution and ii) it yields systems more stable to the choice of learning parameters. A similar study for smoothed n-gram language models was developed in [7]

In section 2 we describe the standard Language Model for ad-hoc document retrieval and in section 3 we propose the new Bayesian predictive model. Section 4 ties the new model to existing ones, and in particular to the two most widely used estimators: linearly interpolated maximum likelihood smoothing and Bayes-smoothing. Finally, Section 5 describes the evaluation of these models.

2. THE UNIGRAM QUERY MODEL

Let us first describe the specific form of Language Model that we consider in this paper, which is exactly the model proposed in [13]. We will refer to this model in the remainder of the paper as the *query unigram model*.

Consider a query $\mathbf{q}$ and a document collection of N documents $C := \{\mathbf{d}_l\}_{l=1,\ldots,N}$, both queries and documents being represented as vectors of indexed term counts:

$$\mathbf{q} \;:=\; (q_1, \ldots, q_i, \ldots, q_V) \in N^V \qquad (1)$$

$$\mathbf{d}_l \;:=\; (d_{l,1}, \ldots, d_{l,i}, \ldots, d_{l,V}) \in N^V \qquad (2)$$

where q_i is the number of times the term i appears in the query and V is the size of the vocabulary.

Furthermore, consider a multinomial generation model for

SIGIR'03, July 28–August 1, 2003, Toronto, Canada.

each document, parameterised by the vector:

$$\theta_l = (\theta_{l,1}, \dots, \theta_{l,i}, \dots, \theta_{l,V}) \in [0,1]^V \ , \ \sum_{i=1}^{V} \theta_{l,i} = 1 \, ,$$

which indicates the probabilities of emission of the different terms in the vocabulary.

Finally, let us also define the *length* of a query ($n_{\mathbf{q}}$) and of a document (n_l) as the sum of their components (e.g. $n_{\mathbf{q}} := \sum_i q_i$).

Under this model, the probability of generating a particular query q with counts $\mathbf{q}$ is given by the product:

$$P(\mathbf{q}|\theta_l) = Z_{\mathbf{q}}^{-1} \prod_{i=1}^{V} (\theta_{l,i})^{q_i} \qquad (3)$$

and, similarly for documents:

$$P(\mathbf{d}_l|\theta_l) = Z_{\mathbf{d}_l}^{-1} \prod_{i=1}^{V} (\theta_{l,i})^{d_{l,i}} \qquad (4)$$

where $Z_{\mathbf{q}} = \begin{pmatrix} n_{\mathbf{q}} \\ q_1, \dots, q_V \end{pmatrix}$ and $Z_{\mathbf{d}_l} = \begin{pmatrix} n_l \\ d_{l,1}, \dots, d_{l,V} \end{pmatrix}$ are constants with respect to the model parameters [1].

The unigram query model postulates that the relevance of a document to a query can be measured by the probability that the *query is generated by the document.* By this it is meant the likelihood of the query (3) when the parameters θ_l are estimated using $\mathbf{d}_l$ as a sample of the underlying distribution.

The central problem of this model is then the estimation of the parameters $\theta_{l,i}$ from the document counts $\mathbf{d}_l$, the collection counts $\left\{cf_i := \sum_{l=1}^{N} d_{l,i}\right\}_{i=1..V}$ and the size of the collection N.

Given an infinite amount of data, (or, in our case, given infinitely long documents), the value of these parameters could be easily estimated by their *empirical estimate*s:

$$\widehat{\theta_{l,i}} = \frac{d_{l,i}}{n_l} \ , \qquad (5)$$

also referred to as the *maximum likelihood estimates* since they maximised the likelihood of the documents as defined in (4) . Unfortunately, if we have of very little data for the estimation of these parameters (in our case, a single smallish document) the empirical estimator is not good: it greatly underestimates the probability of rare words and over-estimates the probability of frequent ones. This is specially dangerous for terms present in the query but not in the document ($\{d_{l,i} = 0 | q_{i \neq 0}\}$, also referred to as *unseen words*) for which the empirical estimate is 0, thus driving to zero the document score regardless of the other matching terms in the query and document.

To alleviate the effect of small data samples, different estimation schools propose different techniques (indeed, different philosophies!) These lead to different adjustments to the empirical estimate (5), often referred to as *smoothed* estimates. Two smoothing techniques have been favoured in the past for Language Models: the maximum-posterior estimator and the linearly-interpolated maximum likelihood estimator. These are discussed in Section 4.

3. BAYESIAN LANGUAGE MODEL

The problem of small data samples and resulting parameter uncertainty suggests the use of Bayesian techniques. Such an approach offers a natural and principled way to take account of uncertainty by *integrating out* the unknown model parameters.

Initially, rather than find a single *point estimate* for the parameter vector θ_l, a *distribution* over θ_l is obtained by combining a *prior* distribution over the model parameters $P(\theta_l)$ with the observation likelihood $P(\mathbf{d}_l|\theta_l)$ using Bayes' rule:

$$P(\theta_l|\mathbf{d}_l) = \frac{P(\theta_l)P(\mathbf{d}_l|\theta_l)}{P(\mathbf{d}_l)} \qquad (6)$$

If the document $\mathbf{d}_l$ is large, then we would expect the *posterior* $P(\theta_l|\mathbf{d}_l)$ to reflect this and to be relatively narrow. In the case of a small document, the posterior would be much broader, thus encapsulating the uncertainty in the value of θ_l.

While the mode of this posterior is exploited by existing maximum-posterior techniques, a more powerful approach is to take account of posterior uncertainty when evaluating the probability of $\mathbf{q}$ by computing the *predictive distribution*:

$$\begin{aligned} P(\mathbf{q}|\mathbf{d}_l) &= \int_{\theta} P(\mathbf{q}|\theta_l) P(\theta_l|\mathbf{d}_l) \, d\theta_l & (7) \\ &= \frac{1}{P(\mathbf{d}_l)} \int_{\theta_l} P(\mathbf{q}|\theta_l) P(\mathbf{d}_l|\theta_l) P(\theta_l) \, d\theta & (8) \end{aligned}$$

where we use the fact that document and query are assumed to be generated by the same distribution[2].

It can be seen that the predictive distribution results from averaging the probability of $\mathbf{q}$ under the model over *all* possible parameter values, weighted by their posterior probability $P(\theta_l|\mathbf{d}_l)$. In the case of abundant data, where the posterior $P(\theta_l|\mathbf{d}_l)$ is peaked around some value $\widehat{\theta}_l$ then it can be seen that $P(\mathbf{q}|\mathbf{d}_l) \approx P(\mathbf{q}|\widehat{\theta}_l)$ and the conventional maximum likelihood predictor, from (5), is recovered. Conversely, when the posterior is broad, the averaging process accounts for the uncertainty in the parameter values.

The choice of prior probability distributions is central to Bayesian inference, especially in the context of small data samples. In theory one should choose a prior distribution that reflects the available knowledge on what makes a *good* solution. In practice, we are restricted to those distributions for which we can compute the integral in (7). In most cases the only available choice for a prior is the *natural conjugate* of the generating distribution, as defined below.

A distribution $P(\theta)$ is called the natural conjugate of the distribution $P(x|\theta)$ if the resulting posterior distribution $P(\theta|x) = P(x|\theta)P(\theta)P(x)^{-1}$ is of the same functional form as the prior $P(\theta)$ [4, section 2.4]. The natural conjugate of a multinomial distribution is the *Dirichlet* distribution:

$$P(\theta) = Z'_{\alpha} \prod_{i=1}^{V} (\theta_i)^{\alpha_i - 1} \qquad (9)$$

with $Z'_{\alpha} = \frac{\Gamma(n_\alpha)}{\prod_{i=1}^{V} \Gamma(\alpha_i)}$ which does not depend on the parameters θ.

We can verify that under this prior the resulting posterior

[1] $\begin{pmatrix} n \\ m_1, \dots m_k \end{pmatrix} := \frac{n!}{\prod_{i=1}^{k} m_i!}$

[2] Alternatively, [14] consider two separate distributions sufficiently similar to make (7) still a valid approximation.

distribution is Dirichelet as well:

$$P(\theta|\mathbf{d}_l) = \frac{\Gamma\left(n_{\mathbf{d}_l} + n_\alpha\right)}{\prod_{i=1}^{V} \Gamma\left(d_{l,i} + \alpha_i\right)} \prod_{i=1}^{V} (\theta_i)^{d_{l,i} + \alpha_i - 1}$$

We will choose (9) as our prior distribution. This distribution has a number of *hyper-parameters* α_i equal to the number of parameters in the model. By virtue of the prior these can be interpreted as additional data or pseudo-counts; we discuss in 3.1 how to set their value.

Finally, using (9) we can compute the predictive distribution in (7) as follows (see detailed derivation in Appendix I):

$$\begin{aligned} P(\mathbf{q}|\mathbf{d}_l) &= Z_{\mathbf{q}} Z'_{\mathbf{d}_l + \alpha} \int \prod_{i=1}^{V} (\theta_i)^{q_i + d_{l,i} + \alpha_i - 1} & (10) \\ &= Z_{\mathbf{q}} \frac{\prod_{i|q_i \neq 0} \prod_{g=1}^{q_i} \left(d_{l,i} + \alpha_i + g - 1\right)}{\prod_{j=1}^{n_{\mathbf{q}}} \left(n_{\mathbf{d}_l} + n_\alpha + j - 1\right)} \,. & (11) \end{aligned}$$

This constitutes our new *document scoring function.* By taking the log and separating out the terms in the query not appearing in the document, we can rewrite it in its final form:

$$\begin{aligned} \log P(\mathbf{q}|\mathbf{d}_l) &= \sum_{i|(q_i, d_i) \neq 0} \sum_{g=1}^{q_i} \log\left(1 + \frac{d_{l,i}}{\alpha_i + g - 1}\right) \\ &- \sum_{j=1}^{n_{\mathbf{q}}} \log\left(n_{\mathbf{d}_l} + n_\alpha + j - 1\right) \\ &+ \sum_{i|q_i \neq 0} \sum_{g=1}^{q_i} \log\left(\alpha_i + g - 1\right) + \log Z_{\mathbf{q}} \quad (12) \end{aligned}$$

where the last two terms can be dropped as they are document independent. We discuss the similarities and differences between this function and the traditional multinomial query model at the end of section 4.

3.1 Setting the hyper-parameter values

The value of the hyper-parameters α depends on our prior knowledge of the problem: before we are given the document $\mathbf{d}_l$, what do we know about the form of the queries generated by it? Several possibilities exist. One is to attribute equal probability to all words in the query, that is, to set all the α_i to some constant. As we will see in section 4 this leads to different maximum-likelihood smoothed estimates.

A better option is to fit the prior distribution to the collection statistics, since these are known. In the absence of any other information, we will assume that the documents (and query) resemble the *average document.* The *average term count* for term t_i is proportional to $P(v_i|C) := \frac{\sum_l d_{l,i}}{\sum_l n_l}$, the prior probability of observing term v_i in a document.

On the other hand, the mean of the posterior distribution in (3) is known to be: $\overline{\theta_i} = \frac{\alpha_i}{n_\alpha}$ [4, Appendix A.2]. Setting this mean to be equal to the average term count we obtain the constraint:

$$\frac{\alpha_i}{n_\alpha} \propto P(v_i|C)$$

which can be satisfied setting $\alpha_i = \mu P(v_i|C)$ and $n_\alpha = \mu$, where μ is a new parameter and is yet undetermined. We will use it as a free parameter in our empirical evaluation of this model.

4. RELATIONSHIP TO OTHER SMOOTHING MODELS

Three types of smoothed maximum likelihood estimators have been used to implement the Language Model: Laplace-smoothing, Bayes-smoothing and linear interpolation smoothing [13][2].

A standard approximation to the Bayesian predictive distribution (7) is the so called *maximum posterior* (MP) distribution. This approximation consists in replacing the integral of the posterior in (7), by its single maximum value:

$$P(\mathbf{q}|\mathbf{d}) \approx P(\mathbf{q}|\theta_l^{MP}) = Z_q^{-1} \prod_{i=1}^{V} \left(\theta_{l,i}^{MP}\right)^{q_i}$$

For a Dirichelet prior, the maximum posterior distribution is known analytically [4]:

$$\theta_{l,i}^{MP} = \frac{d_{l,i} + \alpha_i - 1}{n_l + n_\alpha - V}$$

Setting $\alpha_i = 1$ we obtain the maximum likelihood estimator:

$$\theta_{l,i}^{ML} = \frac{d_{l,i}}{n_l}$$

Setting $\alpha_i = 2$ or more genrally setting $\alpha_i = \lambda + 1$ we obtain the Laplace smoothing estimators:

$$\theta_{l,i}^{LA1} = \frac{d_{l,i} + 1}{n_l + V}$$

$$\theta_{l,i}^{LA\lambda} = \frac{d_{l,i} + \lambda}{n_l + \lambda * V}$$

Finally, setting $\alpha_i = \mu P(v_i|C)$ (as described in section 3.1) we obtain the Bayes-smoothing estimate:

$$\theta_{l,i}^{BS} = \frac{d_{l,i} + \mu P(v_i|C)}{n_l + \mu}$$

We can see from this that these different smoothed estimators are approximations to the full estimator obtained by i) replacing the predictive distribution by the maximum posterior distribution, and ii) choosing a particular value of α[3]

The linear interpolation (LI) smoothing (also referred to as the Jelineck-Mercer smoothing) can be written as:

$$\theta_{l,i}^{LI} = \lambda \theta_{l,i}^{ML} + (1 - \lambda) P(v_i|C) \ .$$

This estimator cannot be derived from the predictive distribution; in fact this type of smoothing can be viewed as a mixture of two generative models.

In *two-stage smoothing* [14] an estimator is developed to combine the Bayes and linear interpolation estimators; this is achieved by replacing θ^{ML} by θ^{BS} in (4). This is a first approximation to the full Bayesian treatment of the linear interpolation smoothing method. However, the full predictive distribution of the mixture model underlying this method does not have an analytical solution, and its treatment is beyond the scope of this paper.

Let us now look at the scoring functions resulting from these estimators. We first note that in the case of BS and LI the probability of an *unseen* word (i.e. $\theta_{l,i}$ when $d_{l,i} = 0$) can

[3] Bayesian statistics is not the only way to derive these estimators. For an alternative treatment see for example [12].

be written as $\beta_l P(v_i|C)$ where β_l is a document dependant constant and $P(v_i|C)$ was previsouly defined. When this is the case we can rewrite the unigram query model (3)[13] as:

$$\log P(\mathbf{q}|\theta_l) \propto_{rank} \sum_{i|\left(q_i, d_{l,i}\right)>0} q_i \log \frac{\theta_{l,i}}{\beta_l P(v_i|C)} + n_{\mathbf{q}} \log \beta_l \tag{13}$$

where the first term only involves query-document matches. If we use any of the other estimators in this section then $\theta_{l,i|dl_i=0} = \beta_l$ and so (13) holds replacing the the last term by β_l. If we use a fall-back smoothing model with any estimator then (13) is always true.

Therefore (13) is quite a general formulation of the unigram query model. Arriving to an equation of this form is crucial for the efficient implementation of the ad-hoc system, for three reasons: i) a fast inverted index can be used to retrieve the weights needed to compute the first term, ii) the number of operations to compute the first term depends only on the number of term-indices matching, and iii) the cost of computing the second term is negligeable. The implementation cost for all the estimators considered here is therefore similar. There exist other estimators that cannot be written in the form of (13), but their implementation in a real IR system is problematic and we have not considered them here.

We note that the Bayesian predictive model proposed in this paper leads to a scoring function similar to (13), excepting that: i) the number of operations to compute the first term in (13) depends now on the number of terms (as oppossed to the number of indices) matching, and ii) the last term cannot be pre-computed since it depends on the query, but its computational cost remains negligible. Therefore, the proposed Bayes predictive model leads to an ad-hoc model only slightly more expensive to compute than the original one, and can be implemented in a real scale IR system.

In fact, the scoring functions resulting from the Bayes predictive distribution and Bayes-smoothing are very similar. Looking at the first term only (the score of matching terms) we can observe that these two functions converge for very long documents and for very short queries, i.e. when uncertainty about the document score is less. The great similarity of the two functions is an indication of how good in practice is the maximum posterior approximation (and thus Bayes smoothing).

5. EMPIRICAL EVALUATION

When measuring the performance of ad-hoc retrieval systems we attempt to estimate the *true* performance of the system, that is, the averaged performance over all the *future* (and therefore unknown) documents and queries. It is well known that the true performance can be arbitrarily far off the performance observed on the *training* collection, that is, the collection being used to optimise the model's parameters. For this reason it is important to differentiate between a *training* and a *test* collection.

In ad-hoc retrieval this differentiation can be tricky. Because document collections tend to be stable or even completely fixed it makes sense to use the same train and test document collections. This is *not* the case however for queries. Therefore the performance of an ad-hoc system must be evaluated on a *test* collection of queries and relevance judgements different from a *training* collection. Most model parameters depend only on the document's counts, and thus will not be affected by the choice of query and relevance judgements set collection. However, the smoothing parameters (and hyper-parameters) do depend on this choice. Therefore, when optimising parameters, we cannot draw conclusions from the performance observed on a particular set of queries.

In order to simulate the process of deciding optimal learning parameter values in a real setting we used the TREC-8 document collection and TREC-6 and TREC-8 queries and query-relevance sets. We produced results for each set of TREC queries separately for a range of smoothing parameters, and then looked at the performance of a set *obtained with the parameter setting optimal to the other set*. This is a crude form of 2-fold cross validation, but has the advantage of producing results comparable to those previously published on TREC collections.

Data pre-processing was standard: terms were stemmed using the Porter stemmer and stop words were removed as well as words appearing fewer than 3 times. Queries were constructed concatenating the title and description of each topic. The TREC-8 document collection was used (TREC-6 relevance judgements about the *Congressional Records* collection were ignored). As well as the experiments reported, we have performed exploratory evalutions using other types of queries (e.g. shorter and longer) and other collections; these experiments yielded results similar to those presented here. Nevertheless an exhaustive evaluation of the proposed model remains to be done. We report two performance measures averaged over queries (i.e. macro-averages): uninterpolated average precision on the top 1000 retrieved documents, and precision after retrieving 10 relevant documents [4] (or after all relevant documents if there were less than 10 in total). For each model, we have indicated in bold underlined the *value obtained on a TREC query set when using the optimal parameter settings obtained from the other TREC query set*.

Tables 1 and 2 present the results obtained for the models discussed previously, for different values of the models parameters and for the two sets of queries. Performance of the ML and Laplace estimators are so much worse than the rest that they are not reported here.

First we note that the new model yields comparable results to the two best reported smoothing models (Bayes-smoothing and the mixture model). Second, we note that in the case of the Bayes predictive model the optimal parameter setting is roughly the same for both query sets and both measures. Average precision peaks at roughly 1K for both sets. Conversely, for Bayes-smoothing this value varies greatly. For example, using $\mu = 2K$ with the Bayes-smoothing model yields excellent results for TREC-8 but poor results for TREC-6. We do not observe this effect in the new Bayes predictive model.

One of the hopes in developing the Bayesian predictive model was to take into account the difference of data uncertainty in short and long documents and in common and rare terms. The fact that this model improves on the Bayes-smoothing model seems to indicate that we have succeeded in doing so, at least to some extent. The gains are even more

[4] We chose this measure instead of the traditional Prec@10 measure to compare the performance of the different estimators in a low-recall regime.

Table 1: Average precision (1E-3) results for the different smoothed estimators.

Bayes-Predictive	$\mu:$	400	600	800	1000	2000
TREC-8		230	234	235	**237**	233
TREC-6		201	214	217	**218**	215
Bayes-Smoothing	$\mu:$	1K	2K	5K	10K	40K
TREC-8		236	241	236	**228**	212
TREC-6		151	**181**	212	223	222
Linearly Interpolated	$\lambda:$	0.01	0.05	0.1	0.2	0.3
TREC-8		214	232	**240**	203	186
TREC-6		159	196	**219**	84	31

Table 2: Precision at recall=10 (1E-3) results for the different smoothed estimators.

Bayes-Predictive	$\mu:$	400	600	800	1000	2000
TREC-8		349	**363**	370	370	355
TREC-6		345	347	**340**	334	330
Bayes-Smoothing	$\mu:$	1K	2K	5K	10K	40K
TREC-8		341	352	344	**333**	320
TREC-6		214	**252**	293	305	294
Linear Interpolation	$\lambda:$	0.01	0.05	0.1	0.2	0.3
TREC-8		337	359	**373**	313	257
TREC-6		276	323	**346**	131	50

impressive for precision at recall=10, as reported in table 2; the new model seems to improve results especially in the low recall region, again indicating that some documents receiving high scores with Bayes-smoothing were successfully demoted by the Bayes predictive model.

Linear interpolation smoothing yields slightly better results than Bayes-smoothing and the Bayes predictive model, and its optimal parameter value also seems quite stable. It has been reported in the past that Bayes-smoothing leads to better performance than linear interpolation smoothing [13], although we have not seen this in our experiments. In these cases, we can expect the Bayes predictive model to produce even higher performances. Furthermore, it has been argued in [13, 14] that linear interpolation smoothing provides an extra level of smoothing more natural to IR tasks; it models background terms appearing in the query. For this reason [14] combined the Bayes-smoothing and linear-interpolation smoothing methods into one, as described in 4. As we have argued earlier, this is not so easily done in the fully Bayesian model; we are currently investigating several approximations to achieve this.

Nevertheless, it seemed interesting to combine the strengths of the Bayes predictive model and linear interpolation smoothing. For this, we adopted a “mixture of experts” approach, and combined the two systems as if they were uncorrelated, by taking the sum of their log scores. For comparison purposes we did this for *every pair* of models: i) Bayes predictive (BP) with linear interpolation (LI), ii) Bayes-smoothing (BS) with LI and iii) BP with BS. Results are shown in table 3. The best absolute results in average precision as well as the best relative gain, are obtained when combining the Bayes predictive model and linear interpolation smoothing. This is again evidence that the strength of the Bayes predictive model (and the derived Bayes smoothing) is complementary to that of linear smoothing, and that they should be combined.

Table 3: Average precision (1E-3) results for pair-wise combinations (score products) of Bayes-smoothing (BS, $\mu = 1K$), Bayes predictive (BP, $\mu = 2K$) and linear interpolation smoothing (LI, $\lambda = 0.1$).

Combiation	TREC-8	TREC-6
BP-LI	257	260
BP-BS	253	250
BS-LI	208	254

6. CONCLUSION

We have presented a first Bayesian analysis of the query unigram Language Model for ad hoc retrieval, and proposed a new scoring function derived from the Bayesian predictive distribution, which many smoothed estimators approximate. Furthermore, we have shown that its computational cost is only slightly greater than that of other existing estimators. Empirical evaluation of this new method shows that the new model performs better than Bayes-smoothing but not as well as linear interpolation smoothing; work remains to be done to combine in an adequate manner these two approaches. An encouraging result in this direction is that a simple combination of their scores produced the best performance. Further work also remains to be done to automatically adapt the μ scaling parameter; we are currently investigating several techniques exploiting the analytical form of the predictive distribution. Finally, we think the Bayesian inference framework could be applied to other language models and could perhaps be used extend this model to other tasks such as relevance feedback, query expansion and adaptive filtering.

7. ADDITIONAL AUTHORS

Additional authors: Stephen Robertson (Microsoft Research Cambridge, U.K.), email: ser@microsoft.com

8. REFERENCES

[1] A Berger and J Lafferty. Information retrieval as statistical translation. In M Hearst, F Gey, and R Tong, editors, *SIGIR’99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 222–229. ACM Press, 1999.

[2] Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Center for Research in Computing Technology. Harvard University, August 1998.

[3] W B Croft, D J Harper, D H Kraft, and J Zobel, editors. *SIGIR 2001: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* ACM Press, 2001.

[4] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis.* Chapman and Hall/CRC, 1885.

[5] D Hiemstra and W Kraaij. Twenty-one at trec-7: ad-hoc and cross-language track. In Voorhees and

Harman [11], pages 227–238. NIST Special Publication 500-242.

[6] V Lavrenko and W B Croft. Relevance-based language models. In Croft et al. [3], pages 120–128.

[7] David J. C. McKay and Linda C. Bauman Peto. A hierarchical dirichlet language model. *Natural Language Engineering*, 1(3):289–307, 1995.

[8] D Miller, T Leek, and R Schwartz. Bbn at trec-7: using hidden markov models for information retrieval. In Voorhees and Harman [11], pages 133–142. NIST Special Publication 500-242.

[9] J M Ponte and W B Croft. A language modeling approach to information retrieval. In W B Croft, A Moffat, C J van Rijsbergen, R Wilkinson, and J Zobel, editors, *SIGIR'98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 275–281. ACM Press, 1998.

[10] S E Robertson and D Hiemstra. Language models and probability of relevance'. In *Proceedings of the first Workshop on Language Modeling and Information Retrieval*, pages 21–25, 2001.

[11] E M Voorhees and D K Harman, editors. *The Seventh Text REtrieval Conference (TREC–7)*. Gaithersburg, MD: NIST, 1999. NIST Special Publication 500-242.

[12] Grace Wahba. *Spline Models for Observational Data*, volume 59. SIAM, 1992.

[13] C Zhai and J Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In Croft et al. [3].

[14] C Zhai and J Lafferty. Two-stage language models for information retrieval. In M Beaulieu, R Baeza-Yates, S H Myaeng, and K Jarvelin, editors, *SIGIR 2002: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 49–56. ACM Press, 2002.

Appendix I: The Dirichlet-Multinomial distribution

$$\begin{aligned}
P(\mathbf{q}|\mathbf{d}_l) &= Z_{\mathbf{q}} Z'_{\mathbf{d}_l+\alpha} \int \prod_{i=1}^{V} (\theta_i)^{q_i+d_{l,i}+\alpha_i-1} \\
&= Z_{\mathbf{q}} \frac{\Gamma(n_{\mathbf{d}_l}+n_\alpha)}{\prod_{i=1}^{V}\Gamma(d_i+\alpha_i)} \frac{\prod_{i=1}^{V}\Gamma\left(q_i+d_{l,i}+\alpha_i\right)}{\Gamma\left(n_{\mathbf{q}}+n_{\mathbf{d}_l}+n_\alpha\right)} \\
&= Z_{\mathbf{q}} \left[\prod_{i|q_i\neq 0} \frac{\Gamma\left(q_i+d_{l,i}+\alpha_i\right)}{\Gamma\left(d_i+\alpha_i\right)}\right] \frac{\Gamma(n_{\mathbf{d}_l}+n_\alpha)}{\Gamma\left(n_{\mathbf{q}}+n_{\mathbf{d}_l}+n_\alpha\right)} \\
&= Z_{\mathbf{q}} \frac{\prod_{i|q_i\neq 0}\prod_{g=1}^{q_i}\left(d_{l,i}+\alpha_i+g-1\right)}{\prod_{j=1}^{n_{\mathbf{q}}}\left(n_{\mathbf{d}_l}+n_\alpha+j-1\right)}
\end{aligned}$$

In the last line we used the following property: $\Gamma(x+n) = (x+n-1)\cdots(x+1)\Gamma(x+1)$.