Information Retrieval Using Hierarchical Dirichlet Processes

Philip J. Cowans
Inference Group, University of Cambridge
Cavendish Laboratory, Madingley Road, Cambridge, CB3 0HE, UK
pjc51@cam.ac.uk

ABSTRACT

An information retrieval method is proposed using a hierarchical Dirichlet process as a prior on the parameters of a set of multinomial distributions. The resulting method naturally includes a number of features found in other popular methods. Specifically, tf.idf-like term weighting and document length normalisation are recovered. The new method is compared with Okapi BM-25 [3] and the Twenty-One model [1] on TREC data and is shown to give better performance.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*retrieval models*.

General Terms: Algorithms, Theory.

Keywords: Hierarchical Dirichlet processes, probabilistic information retrieval.

1. INTRODUCTION

Given a collection $\mathcal{C}$ consisting of a number of documents, $\{\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots\}$ and a query $\boldsymbol{q}$, the task of an information retrieval method is to return a list of documents ordered by relevance to $\boldsymbol{q}$.

Each document in the collection consists of a number of terms, denoted by x. Let y be a label associated with each term to indicate the document from which it was taken. The whole collection can now be viewed as a set of (x, y) pairs, which can be viewed as samples from a probabilistic model. To perform information retrieval, a label is also associated with each term in the query. It is assumed that the query (x, y) pairs were generated from the same model as those in the collection, but with the additional constraint that the query labels must all take the same value, y_q. The relevance of document $\boldsymbol{d}$ to $\boldsymbol{q}$, $R(\boldsymbol{d}, \boldsymbol{q})$, can now be defined as the logarithm of the probability that $y_q = y_d$, the label corresponding to $\boldsymbol{d}$. The symbols $\boldsymbol{x}_\mathcal{C}$ and $\boldsymbol{y}_\mathcal{C}$ will be used to denote vectors containing the x and y values for all terms in the collection, and $\boldsymbol{x}_q$ denotes the vector of terms in the query.

Predictions for y_q can be made using Bayes' rule,

$$\Pr(y_q \mid \boldsymbol{x}_q, \boldsymbol{y}_\mathcal{C}, \boldsymbol{x}_\mathcal{C}) \propto \Pr(\boldsymbol{x}_q \mid y_q, \boldsymbol{y}_\mathcal{C}, \boldsymbol{x}_\mathcal{C}) \cdot \Pr(y_q \mid \boldsymbol{y}_\mathcal{C}, \boldsymbol{x}_\mathcal{C})$$

The prior, $\Pr(y_q \mid \boldsymbol{y}_\mathcal{C}, \boldsymbol{x}_\mathcal{C})$ can be used to incorporate additional information, for example from an analysis of link structure in a hypertext collection. In this work however, a uniform prior will be used. It will also be assumed that query terms are independent given the collection, which gives

$$\Pr(y_q \mid \boldsymbol{x}_q, \boldsymbol{y}_\mathcal{C}, \boldsymbol{x}_\mathcal{C}) = \frac{1}{Z} \prod_i \Pr\left(x_q^{(i)} \mid y_q, \boldsymbol{y}_\mathcal{C}, \boldsymbol{x}_\mathcal{C}\right)$$

where $x_q^{(i)}$ is the i^{th} term in the query and Z is a normalising constant. The relevance as defined above is therefore

$$R(\boldsymbol{d}, \boldsymbol{q}) \approx \sum_i \log\left(\Pr\left(x_q^{(i)} \mid y_q = y_d, \boldsymbol{y}_\mathcal{C}, \boldsymbol{d}_\mathcal{C}\right)\right) \qquad (1)$$

where constant terms have been omitted.

It is possible to express the language modelling approach to information retrieval [2] within this framework by using

$$\Pr\left(x_q^{(i)} \mid y_q, \boldsymbol{y}_\mathcal{C}, \boldsymbol{x}_\mathcal{C}\right) = P_{y_q}\left(x_q^{(i)}\right)$$

where $P_{y_d}(x)$ is the language model corresponding to document $\boldsymbol{d}$. In this work a slightly different approach is used. Instead of using a separate language model for each of the documents, a single model is used for the whole collection which generates terms conditioned on the document label. This allows information to be shared between documents in a principled way. Section 2 of this paper describes the model used and Section 3 presents an evaluation of the model on TREC data. Section 4 discusses the new model as well as possible future extensions. Finally, Section 5 summarises the conclusions of this work.

2. THE MODEL

The proposed model consists of a separate multinomial distribution over terms for each label. The multinomial corresponding to label y has parameter $\boldsymbol{m}_y$. Documents are constructed by making independent draws of terms from the multinomial corresponding to the appropriate label. A hierarchical Dirichlet process prior is used for the multinomial parameters. In other words, the parameters $\boldsymbol{m}_y$ are themselves drawn from a Dirichlet distribution with parameter $\lambda_1 \boldsymbol{m}$. The use of a single parameter, $\boldsymbol{m}$, allows information to be shared between the documents. The prior distribution for $\boldsymbol{m}$ is another Dirichlet distribution with parameter $\lambda_2 \boldsymbol{u}$, where $\boldsymbol{u}$ is a uniform distribution over terms. λ_1 and λ_2 are model parameters. To summarise,

$$\begin{aligned} x_i &\sim \text{Multinomial}(\boldsymbol{m}_{y_i}) \\ \boldsymbol{m}_y &\sim \text{Dirichlet}(\lambda_1 \boldsymbol{m}) \\ \boldsymbol{m} &\sim \text{Dirichlet}(\lambda_2 \boldsymbol{u}) \end{aligned}$$

SIGIR'04, July 25–29, 2004, Sheffield, South Yorkshire, UK.
ACM 1-58113-881-4/04/0007.

It can be shown that samples from this distribution may be obtained by using an *oracle* shared between the labels [4]. For each label a count is kept of the number of times that each term has been drawn before in the context of that label. When asked to produce a new term conditioned on a document label, each term is returned with probability proportional to the corresponding count. For each document a fixed number of additional counts equal to λ_1 are reserved to represent asking the oracle for a sample. The oracle also maintains a count of how many times each term has been returned when it was consulted in the past, and returns a sample distributed proportionally to this count. Again, a fixed number of counts, this time equal to λ_2 are reserved, which in this case correspond to making predictions uniformly over all possible terms. Hence predictions can be made even if no data has been seen before.

As shown in (1), determining the relevance of a document involves predicting $\boldsymbol{x}_q$ conditioned on y_q, $\boldsymbol{x}_\mathcal{C}$ and $\boldsymbol{y}_\mathcal{C}$. In order to do this it is not only necessary to know the term counts for each document, but also whether the oracle was asked when producing each sample as this determines the counts used by the oracle. A full evaluation would marginalise over all such 'paths' through the oracle, but as the number of these grows exponentially with the collection size, this approach rapidly becomes intractable. Fortunately, good results are obtained by conditioning on a particular choice of path; that where the oracle is only ever asked the first time that a term is seen in each document.

Using this approximation gives query term predictions of the form

$$\Pr(x \mid y) = \frac{1}{N_{y_d} + \lambda_1} \left(\mathsf{tf}(x, y) + \lambda_1 \hat{\mathrm{p}}(x) \right)$$

where $\mathsf{tf}(x, y)$ is the term frequency, or the number of times that term x appears in the document with label y and N_{y_d} is the length of document $\boldsymbol{d}$. $\hat{\mathrm{p}}(x)$ is given by

$$\hat{\mathrm{p}}(x) = \frac{\mathsf{df}(x) + \frac{\lambda_2}{N}}{\sum_{x'} \mathsf{df}(x') + \lambda_2}$$

where $\mathsf{df}(x)$ is the document frequency, i.e. the number of documents containing x at least once and N is the total number of different terms in the collection. Note that the use of $\mathsf{df}(x)$ arises naturally from the oracle counts.

The log probability of the whole query is

$$\log \Pr(\boldsymbol{x}_q \mid y_q) = \sum_i \log \left(\frac{1}{N_{y_q} + \lambda_1} \right) + \sum_i \log \left(1 + \frac{\mathsf{tf}\left(x_q^{(i)}, y_q\right)}{\lambda_1 \hat{\mathrm{p}}\left(x_q^{(i)}\right)} \right) + \sum_i \log \left(\lambda_1 \hat{\mathrm{p}}\left(x_q^{(i)}\right) \right)$$

The last term in this expression is not dependant on y_q, and may be ignored for the purposes of ranking. The relevance can therefore be written as

$$R(\boldsymbol{d}, \boldsymbol{q}) = \sum_i \log \left(1 + \frac{\mathsf{tf}\left(x_q^{(i)}, y_d\right)}{\lambda_1 \hat{\mathrm{p}}\left(x_q^{(i)}\right)} \right) + N_q \log \left(\frac{1}{N_{y_d} + \lambda_1} \right)$$

where N_q is the length of the query. The first term in this expression provides tf.idf-like term weighting. The second term can be interpreted as providing global document length normalisation.

	TREC-7	TREC-8
BM-25	0.2145	0.2481
Twenty-One	0.2224	0.2621
Dirichlet	*0.2335*	*0.2701*

Table 1: Average non-interpolated precision scores over top 1000 documents for TREC-7 and -8 *ad-hoc* tasks. The Dirichlet results used $\lambda_1 = 1250$ and $\lambda_2 = 750$. BM-25 used $k_1 = 1.2$, $k_3 = 7$ and $b = 0.75$ and the Twenty-One model used $\alpha_1 = 0.85$ and $\alpha_2 = 0.15$.

3. RESULTS

The model was tested on the *ad-hoc* tasks from TREC-7 and TREC-8, using all data on discs 4 and 5 except for the CR texts and with queries 351-400 and 401-450 respectively. The full query text consisting of the title, description and narrative was used in all cases. The only pre-processing that was done prior to indexing was basic stop-word removal and stemming using a Porter stemmer. The experiments were performed using the LEMUR language modelling toolkit. As well as the Dirichlet model, results were obtained for the Okapi BM-25 [3] and the Twenty-One consortium models. Results are shown in Table 1. The results show that the Dirichlet model gives better performance in these tasks.

4. DISCUSSION

This model is in many respects similar to the model proposed by the Twenty-One consortium [1]. Differences occur in the form of document length normalisation and in the fact that the Twenty-One model is also restricted to raw document frequency values (corresponding to $\lambda_2 = 0$). However, perhaps the most important difference is that the use of document frequency information must be postulated *ab initio* in the Twenty-One model, whereas it arises naturally in the derivation of the Dirichlet model.

A key advantage of the Dirichlet approach is the potential to extend the model in a principled fashion. Such extensions might include incorporation of bi-gram and higher order statistics, the use of a hierarchical document classification (e.g. by author and/or source) and the use of meta-data such as hypertext link text in web collections.

5. CONCLUSION

This work proposes a new probabilistic information retrieval method based on a hierarchical Dirichlet process prior. The proposed model naturally reproduces features found in other methods, such as tf.idf weighting an document length normalisation, and outperforms BM-25 and the Twenty-One model in two TREC *ad-hoc* tasks.

6. REFERENCES

[1] D. Hiemstra and W. Kraaij. Twenty-one at TREC-7: Ad-hoc and cross-language track. In *Text REtrieval Conference*, pages 174–185, 1998.

[2] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Research and Development in Information Retrieval*, pages 275–281, 1998.

[3] S. E. Robertson, S. Walker, M. Hancock-Beaulieu, A. Gull, and M. Lau. Okapi at TREC. In *Text REtrieval Conference*, pages 21–30, 1992.

[4] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical dirichlet processes. Technical Report 653, Department Of Statistics, UC Berkeley, 2003.