Using Language Models and Topic Models for XML Retrieval

Fang Huang

School of Computing, The Robert Gordon University, Scotland
f.huang@rgu.ac.uk

Abstract. This paper exposes the results of our participation in the INEX 2007 ad hoc track. We implemented two different models: a mixture language model and a topic model. For the language model, we focused on the question of how shallow features of text display information in an XML document can be used to enhance retrieval effectiveness. Our language model combined estimates based on element full-text and the compact representation of the element. We also used non-content priors, including the location the element appears in the original document, and the length of the element path, to boost retrieval effectiveness. For the topic model, we looked at a recent statistical model called Latent Dirichlet Allocation[1], and explored how it could be applied to XML retrieval.

1 Introduction

In this paper, we describe our experiments in the INEX 2007 ad hoc track. With the rapidly widespread use of the eXtensible Markup Language (XML) on the internet, XML information retrieval (XML-IR) has been receiving growing research interest. A variety of approaches have been exploited to score XML elements' relevance to a user's query. Geva [3] described an approach based on the construction of a collection sub-tree that consists of all elements containing one or more of the query terms. Leaf nodes are assigned a score using a *tf.idf* variant, and scores are propagated upwards in the document XML tree, so that all ancestor elements are ranked. Ogilvie and Callan [7] proposed using hierarchical language models for ranking XML elements. An element's relevance is determined by weighted combining of several language models estimated, respectively, from the text of the element, its parent, its children, and the document. In our participation of INEX 2006, we[4] investigated which parts of a document or an XML element are more likely to attract a reader's attention, and proposed using these "attractive" parts to build a compact form of a document (or an XML element). We then used a mixture language model combining estimates based on element full-text, the compact form of it, as well as a range of non-content priors. The mixture language model presented in this paper is mainly based on our previous approach[4], but we made a few modifications to improve retrieval effectiveness.

N. Fuhr et al. (Eds.): INEX 2007, LNCS 4862, pp. 94–102, 2008.
© Springer-Verlag Berlin Heidelberg 2008

We also experimented on how topic model, a recent unsupervised learning technique, can be use in XML retrieval. The specific model at the heart of this study is the Latent Dirichlet Allocation (LDA) model[1], a hierarchical Bayesian model employed previously to analyze text corpora and to annotate images[2]. The basic idea of a topic model is that documents are mixtures of topics, where a topic is a probability distribution over words. We used LDA to discover topics in the Wikipedia collection. Documents, XML elements, user queries and words were all represented as mixtures of probabilistic topics, and were compared to each other to calculate their relevance.

The remainder of this paper is organized as follows: Section 2 describes the mixture language model we used. Section 3 briefly introduces the LDA model and explains how LDA is used to model the relationships of documents in the Wikipedia collection. Our INEX experiments and submitted runs are presented in section 4. Section 5 discusses our results in the INEX 2007 official evaluation. The final part, section 6, concludes with a discussion and possible directions for future work.

2 The Retrieval Model

While current work in XML information retrieval focuses on exploiting the hierarchical structure of XML elements to implement more focused retrieval strategies, we believe that text display information together with some shallow features (e.g., an XML element's location in the original document) could be used to enhance retrieval effectiveness. This is based on the fact that when a human assessor reads an article, he (or she) usually can judge its relevance by skimming over certain parts of the documents. Intuitively, the titles, section titles, figures, tables, words underlined, and words emphasized in bold, italics or larger fonts are likely to be the most representative parts. In [4], we proposed to extract and put together all those most representative words to build a compact form of a document (or an XML element), and employed retrieval models that emphasized the importance of the compact form in identifying the relevance of an XML element. However, our results in the INEX 2006 evaluation showed that it did not perform as well as we expected. One reason might be that a compact form built like that contained some noise, as in the large, heterogeneous collection we used, not all the features we used are related to texts' importances. Based on this consideration, in this work, the compact form was generated by words only from titles, section titles, and figure captions. For the remainder of the paper, when we refer to the compact form of an XML element, we mean a collection of words extracted from the titles, section titles, and figure captions nested within that element.

The retrieval model we used is based on the language model, i.e., an element's relevance to a query is estimated by

$$P(e|q) \propto P(e) \cdot P(q|e) \tag{1}$$

where e is an XML element; q is a query consisting of the terms t_1,...,t_k; the prior, $P(e)$, defines the probability of element e being relevant in absence of a query; $P(q|e)$ is the probability of the query q, given element e.

2.1 Element Priors

The prior $P(e)$ defines the probability that the user selects an element e without a query. Elements are not equally important even though their contents are ignored. Several previous studies[5,9] reported that a successful element retrieval approach should be biased towards retrieving large elements. In INEX 2006, we conducted a preliminary experiment to investigate potential non-content features that might be used to boost retrieval effectiveness, and concluded that relevant elements tend to appear in the beginning parts of the text, and they are not likely to be nested in depth[4].

Based on these considerations, we calculate the prior of an element according to its location in the original document, and the length of its path.

$$P(e) = \frac{1}{5 + |e_{location}|} \cdot \frac{1}{3 + |e_{path}|} \tag{2}$$

where $e_{location}$ is the location value of element e; and e_{path} is the path length of e. Location was defined as the local order of an element ignoring its path. The path length of an element e equals to the number of elements in the path including e itself and those elements nesting e. For example, for an element /article[1]/body[1]/p[1] (the first paragraph in the document), the location value is 1 (the first paragraph), and the path length is 3.

2.2 Probability of the Query

Assuming query terms to be independent, $P(q|e)$ can be calculated according to a mixture language model:

$$P(q|e) = \prod_{i=1}^{k} (\lambda \cdot P(t_i|C) + (1 - \lambda) \cdot P(t_i|e)) \tag{3}$$

where λ is the so-called smoothing parameter; C represents the whole collection. $P(t_i|C)$ is the estimate based on the collection used to avoid sparse data problem.

$$P(t_i|C) = \frac{doc\_freq(t_i, e)}{\sum_{t' \in C} doc\_freq(t', C)} \tag{4}$$

The element language model, $P(t_i|e)$, defines where our method differs from other language models. In our language model, $P(t_i|e)$ is estimated by a linear combination of two parts:

$$P(t_i|e) = \lambda_1 \cdot P(t_i|e_{full}) + (1 - \lambda - \lambda_1) \cdot P(t_i|e_{compact}) \tag{5}$$

where λ_1 is a mixture parameter; $P(t_i|e_{full})$ is a language model for the full-text of element e; $P(t_i|e_{compact})$ is the estimate based on the compact representation of element e. Parameter λ and λ_1 play important roles in our model. Previous experiments[5,10] suggested that there was a correlation between the value of the smoothing parameter and the size of the retrieved elements. Smaller average sizes of retrieved elements require more smoothing than larger ones. In our experiments, the retrieval units, which are XML elements, are relatively small. We set the smoothing parameter $\lambda = 0.6$. And λ_1 was set to 0.3. In summary, the probability of a query is calculated by

$$P(q|e) = \prod_{i=1}^{k}(0.6(t_i|C) + 0.3(t_i|e_{full}) + 0.1(t_i|e_{compact})). \quad (6)$$

3 Using the Latent Dirichlet Allocation Model on Wikipedia Collection

Latent dirichlet allocation[1] is a generative probabilistic model for collections of discrete data such as text corpora. It assumes that each word of each document is generated by one of several "topics"; each topic is associated with a different conditional distribution over a fixed vocabulary. The same set of topics is used to generate the entire set of documents in a collection but each document reflects these topics with different relative proportions. Specifically, for a collection consists of words $w = w_1, w_2, ..., w_n$, where $w_i(1 \leq i \leq n)$ belongs to some documents, as in a word-document co-occurrence matrix. For each document d_i, we have a multinomial distribution over k topics, with parameters $\theta^{(d_i)}$, so for a word in document d_i, $P(z_i = j) = \theta_j^{(d_i)}$. The $j^{th}(1 \leq j \leq n)$ topic is represented by a multinomial distribution over the n words in the vocabulary, with parameters $\alpha^{(j)}$, so $P(w_i|z_i = j) = \alpha_{w_i}^{(j)}$. A Dirichlet prior is introduced for the topic distribution with parameters $\alpha_i(1 \leq i \leq k)$:

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{i=1}^{k}\alpha_i)}{\prod_{i=1}^{k}\Gamma(\alpha_i)}\theta_1^{\alpha_1-1}...\theta_k^{\alpha_k-1} \quad (7)$$

where the parameter α is a k-vector with components $\alpha_i > 0$, and $\Gamma(x)$ is the Gamma function. Thus, the probability of observing a document d_i is:

$$p(d_i|\alpha, \beta) = \int p(\theta|\alpha)(\prod_{n=1}^{N}\sum_{z_n} p(z_n|\theta)p(w_n|z_n, \beta))d\theta \quad (8)$$

where document d_i contains N words $w_n(1 \leq n \leq N)$. The number of parameters to estimate in this model is k parameters for the Dirichlet distribution and $n-1$ parameters for each of the k topic models. The estimation of parameters is done by variational inference algorithms.

We applied the LDA on the Wikipedia collection. All texts in the collection were lower-cased, stop-words removed using a stop-word list. After the preprocessing, each document was represented in a form of a word frequency vector.

A Gibbs sampling algorithm was then used to estimate parameters of LDA in our implementation. As the LDA model assumes that the dimensionality of the Dirichlet distribution (and thus the dimensionality of the topic variable z) is known and fixed, two topic models were learned in our experiments. The dimensionalities of them were 200 and 50, respectively. The content of words, documents, any XML elements, and user queries were then represented as vectors of topic probabilities. The similarity of a user query and an XML element were determined by cosine similarity between the two corresponding vectors.

4 INEX Experiments

In this section, we present our experiments in participating for the INEX 2007 ad hoc track.

4.1 Index

We created inverted indexes of the collection using Lucene[6]. Indexes were word-based. All texts were lower-cased, stop-words removed using a stop-word list, but no stemming. We considered paragraph elements to be the lowest possible level of granularity of a retrieval unit, and indexed text segments consisting at least one paragraph as a descendant element. For the remainder of the paper, when we refer to the XML elements considered in our investigation, we mean the segments that correspond to paragraph elements and to their ancestors. For each XML element, all text nested inside it was indexed. In addition to this, we added an extra field which corresponded to the compact representation of the element. As some studies[5,9] have already concluded that a successful element retrieval approach should be biased toward retrieving large elements, in the experiments, we indexed only those elements that consist of more than 200 characters (excluding stop words). The decision to measure in characters instead of words was based on the consideration that smaller segments such as "I like it." contains little information, while a sentence with three longer words tends to be more informative.

4.2 Query Processing

Our queries were created using terms only in the <title> parts of topics. Like the index, queries were word-based. The text was lower-cased and stop-words were removed, but no stemming was applied. '+', '-' and quotes in queries were simply removed. The modifiers "and" and "or" are ignored.

4.3 Submissions

We totally submitted 9 runs for the ad hoc track, three for each of the 3 tasks (Focused, Relevant-in-Context, and Best-in-Context). Table 1 lists a brief description of the runs.

Table 1. Ad-hoc runs submitted to INEX'07

RunID	Approach	INEX task
Focused-LM	mixture language model	Focused
Focused-TM-1	topic model with 200 topics	Focused
Focused-LDA	topic model with 50 topics	Focused
RelevantInContent-LM	mixture language model	Relevant-in-Context
RelevantInContent-TM-1	topic model with 200 topics	Relevant-in-Context
RelevantInContent-LDA	topic model with 50 topics	Relevant-in-Context
BestInContext-LM	mixture language model	Best-in-Context
BestInContext-TM-1	topic model with 200 topics	Best-in-Context
BestInContext-LDA	topic model with 50 topics	Best-in-Context

In our experiments, the top ranked elements were returned for further processing. For the Focused task, overlaps were removed by applying a post-filtering on the retrieved ranked list by selecting the highest scored element from each of the paths. In case of two overlapping elements with the same relevance score, the child element was selected. For the Relevant-in-Context task, we simply took the results for the Focused task, reordered the elements in the list such that results from the same article were grouped together in the same order they appeared in the original article. In the Best-in-Context task, the element with the highest score was chosen for each document. If there were two or more elements with the same highest score, the one that appears first in the original document was selected. For each of the runs, the top 1,500 ranked elements were returned as answers.

5 Evaluation and Results

The system's performance was evaluated against the INEX human relevance assessments. Details of the evaluation metrics can be found in [8]. Table 2 lists the result of our Focused runs, where iP@j, $j \in [0.00, 0.01, 0.05, 0.10]$, is the interpolated precision at j recall level cutoffs, and MAip is the mean average interpolated precision. Evaluation results of Relevant-in-Context runs and Best-in-Context runs are listed in Table 3 and Table 4, respectively. Here, g[r], $r \in [5, 10, 25, 50]$, is non-interpolated generalized precision at r ranks; and MAgP is non-interpolated mean average generalized precision.

In general, our method based on mixture language model performed well compared to other submissions. Due to the pressure of time, we did not submit baseline runs for retrieval models based on full-text solely or without priors for comparison. Performances of Focused-LDA, RelevantInContext-LDA, and BestInContext-LDA are very poor. This is what we expected, as we used only 50 topics to model the collection in this group of runs. The results prompt us that 50 topics are not enough to describe the whole collection. This is reasonable, as the Wikipedia collection we used is a large heterogeneous corpus containing 659,388 documents with a large number of various topics. Furthermore, when we increased the number of topics,

Table 2. Results of Focused runs (totally 79 submissions)

RunID	*i*P@0.00		*i*P@0.01		*i*P@0.05		*i*P@0.10		MAiP	
	score	rank	score	rank	score	rank	score	rank	score	rank
Focused-LM	0.5120	33	0.4758	22	0.4118	13	0.3803	13	0.1894	8
Focused-TM-1	0.5346	18	0.4711	27	0.3788	34	0.3157	37	0.1301	31
Focused-LDA	0.0564	79	0.0277	79	0.0216	78	0.0188	78	0.0066	78

Table 3. Results of Relevant-in-Context runs (totally 66 submissions)

RunID	gP[5]		gP[10]		gp[25]		gp[50]		MAgP	
	score	rank	score	rank	score	rank	score	rank	score	rank
RelevantInContext-LM	0.2531	8	0.2205	12	0.1680	13	0.1283	14	0.1302	12
RelevantInContext-TM-1	0.2299	21	0.2064	18	0.1598	18	0.1270	17	0.1189	19
RelevantInContext-LDA	0.0100	66	0.0074	66	0.0122	65	0.0102	65	0.0081	63

Table 4. Results of Best-in-Context runs (totally 71 submissions)

RunID	gP[5]		gP[10]		gp[25]		gp[50]		MAgP	
	score	rank	score	rank	score	rank	score	rank	score	rank
BestInContext-LM	0.3405	5	0.2906	4	0.2278	4	0.1761	5	0.1742	8
BestInContext-TM-1	0.2273	39	0.2129	41	0.1775	35	0.1402	35	0.1308	35
BestInContext-LDA	0.0126	69	0.0091	69	0.0114	69	0.0099	69	0.0093	69

performances of Focused-TM-1, RelevantInContext-TM-1, and BestInContext-TM-1 (runs based on a topic model with 200 topics) are significantly improved. As the topic dimensionalities were randomly set as 50 and 200 in our experiments, we expect that retrieval results will be significantly improved given that we know the actually number of topic underlying the collection.

6 Conclusions and Future Work

We have presented, in this paper, our experiments of using language models and topic models for the INEX 2007 evaluation campaign. In our language model, we assumed important words could be identified according to the ways they were displayed in the text. We proposed to generate a compact representation of an XML element by extracting words appearing in titles, section titles, and figure captions the element nesting. Our retrieval methods emphasized the importance of these words in identifying relevance. We also integrated non-content priors that emphasized elements appeared in the beginning part of the original text, and elements that are not nested deeply. We used a mixture language model combining estimates based on element full-text, the compact form of it, as well as the non-content priors. In general, our system performed well compared to other submissions. However, due to the pressure of time, we could not submit

baseline runs for comparisons of exactly how these priors and compact forms improve performances.

Our future work will focus on refining the retrieval models. Currently, the compact representation of an element is generated by words from certain parts of the text. However, the effectiveness of this method depends on the type of the documents. For example, in scientific articles, section titles (such as introduction, conclusion, etc) are not very useful for relevance judgment, whereas section titles in news reports are very informative. In the future, we will explore different patterns for generating compact representations depending on types of texts. This might involve genre identification techniques. We will investigate different priors' effectiveness and how different types of evidence can be combined to boost retrieval effectiveness.

We also explored how topic models can be used in XML retrieval. The LDA model was used to detect topics underlying the collection. We learned two topic models with topic numbers of 50 and 200, respectively. The evaluation results showed that runs based on the topic model with 200 topics achieved significantly better performances than runs based on a lower-dimensional topic space (50 topics). One assumption of the LDA model is that the dimensionality of the topic is known and fixed. In our experiments, dimensionalities were randomly set as 50 and 200. We expect the results will be better if we learn the number of topics underlying the collection. Our future work will focus on integrating text mining techniques to learn the number of topics before applying LDA model.

Acknowledgments

The Lucene-based indexer used this year was partly based on the indexing code developed for RGU INEX'06 by Stuart Watt and Malcolm Clark.

References

1. Blei, D., Ng, A., Jordan, M.: Latent Dirichlet Allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
2. Blei, D., Jordan, M.: Modeling annotated data. In: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 127–134. ACM Press, New York (2003)
3. Geva, G.: Gardens point XML IR at INEX 2005. In: Proceedings of Fourth Workshop of the INitiative for the Evaluation of XML Retrieval (INEX 2005) (2006)
4. Huang, F., Watt, S., Harper, D., Clark, M.: Compact representations in XML retrieval. In: Fuhr, N., Lalmas, M., Trotman, A. (eds.) INEX 2006. LNCS, vol. 4518. Springer, Heidelberg (2007)
5. Kamps, J., Marx, M., de Rijke, M., Sigurbjornsson, B.X.: Retrieval: What to retrieve?. In: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (2003)
6. Lucene. The Lucene search engine (2005), `http://jakarta.apache.org/lucene`
7. Ogilvie, P., Callan, J.: Parameter estimation for a simple hierarchical generative model for XML retrieval. In: Proceedings of Fourth Workshop of the INitiative for the Evaluation of XML Retrieval (INEX 2005) (2006)

8. Pehcevski, J., Kamps, J., Kazai, G., Lalmas, M., Ogilvie, P., Piwowarski, B., Robertson, S.: INEX 2007 Evaluation Measures. In: INEX 2007 (2007)
9. Sigurbjornsson, B., Kamps, J., de Rijke, M.: An element-based approach to XML retrieval. In: INEX 2003 Workshop Proceedings (2004)
10. Zhai, C., Lafferty, J.: 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 (2001)