Matching Resumes and Jobs Based on Relevance Models

Xing Yi, James Allan and W. Bruce Croft
Center for Intelligent Information Retrieval, Department of Computer Science
140 Governor's Drive, University of Massachusetts, Amherst, MA 01003-4610, USA
{yixing,allan,croft}@cs.umass.edu

ABSTRACT

We investigate the difficult problem of matching semi-structured resumes and jobs in a large scale real-world collection. We compare standard approaches to Structured Relevance Models (SRM), an extension of relevance-based language model for modeling and retrieving semi-structured documents. Preliminary experiments show that the SRM approach achieved promising performance and performed better than typical unstructured relevance models.

Categories and Subject Descriptors: H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval

General Terms: Algorithms, Experimentation

Keywords: Relevance Models, Resume, Job Matching

1. INTRODUCTION

We are interested in finding resumes that are appropriate matches to a job description, where *appropriate* means that a prospective employer would be interested in reading the retrieved resumes. We carry out a series of experiments on a dataset consisting of over a million resumes, almost a quarter million job descriptions, and a large number of relevance judgments that indicate which resumes are potentially interesting for a particular job description.

Prospective employees or employers usually submit their resume or job information through online forms that contain many free text fields such as *job title*, *biography*, etc. This information is typically maintained by a relational database engine. An ideal system would retrieve candidate resumes for a job or a list of jobs potentially suitable for a candidate. However using a relational engine for this matching task will run into two major obstacles. First, many fields are input as free form text by users rather than a set of agreed upon keywords from a closed vocabulary. That means that the contents cannot be reliably predicted; the problem is more of a classic information retrieval one. A second obstacle is that many fields are missing: users often do not input all the fields in an online form. For example, in our collection, 23% of the resumes do not have a *ResumeBody* field and 90% are missing the *Summary* field.

Our primary approach for this problem is Structured Relevance Models (SRM) [3], a retrieval model for semi-structured documents based on the idea that plausible values for a given field could be inferred from the context provided by the other fields in the record. For instance, if two jobs have "Database Administrator" in the *JobTitle* fields, it is likely that appropriate candidates' resumes for both jobs should have "SQL server" or "MySQL" in the *ResumeBody* fields. We will formally describe this matching resume/job task and then present different approaches for it. Then we will describe some preliminary experimental results by applying these different approaches.

SIGIR'07, July 23–27, 2007, Amsterdam, The Netherlands.
ACM 978-1-59593-597-7/07/0007.

2. DIFFERENT APPROACHES

Given a collection of semi-structured resumes R, a collection of semi-structured jobs J, and some known matched resume/job pairs $< \mathbf{r}, \mathbf{j} >$. The task is to retrieve a list of related resumes for any existing or new job $\mathbf{j}$, or retrieve a list of related jobs for any existing or new resume $\mathbf{r}$. We will focus on the former this task here; the other direction could be done in similar way and has similar performance.

Our two baseline runs ignore the structure of the documents. To do that, we strip the structure from the resume and job records, flattening the data by concatenating the free form text in all the fields. We use a query likelihood approach, with the flattened job record as the query and the flattened resumes as the documents. We call this simple language modeling approach "sLM". We expect its performance to be weak because it does not have any way to bridge the vocabulary divide between job descriptions and resumes–e.g., the DB administrator and SQL server example above.

To address that problem, our second baseline run ignores the structure but leverages past judged pairs in a type of supervised query expansion. This approach is a variation of Relevance Models [2] where the relevance model is built from known relevant documents (resumes) rather than from highly ranked ones. We call this approach tRM for "true relevance model." It runs in three steps: (1) we run the flattened job record as a query against the flattened job collection, and retrieve a list of similar jobs; (2) we utilize the resumes are known (by our relevance judgments) to be related to those retrieved jobs, and build a relevance language model from them; and (3) we run the relevance model against the flattened resume collection and retrieve a list of similar resumes. Note that this approach has the opportunity to bring resume-specific language that is related to the job into the query.

Our final model is the SRM approach [3]. It uses relevance information in the same way that tRM does, but it also uses the structure of the fields as well as their inter-dependence.

	records covered	average length	unique terms
ResumeTitle	1,276,566	3	92,403
ResumeBody	988,107	477	1,636,980

Table 1: Statistics for some textual fields.

It follows roughly the same three steps, but operates slightly differently because of the multiple fields. For SRM we first run *each* field of a given job **j** as a query against the corresponding field of the semi-structured job collection J, and merge the field-specific retrieved jobs using weighted cross-entropy [1]. We retain only the top k most highly ranked jobs. As with tRM, we now have a set of jobs that are similar to the query job; in contrast to tRM we used the field structure of the jobs to find them. In our second step, we use the resumes known to be related to the retrieved jobs and build relevance models, but this time we build one model per field in the resumes. In the third step, we run each of those field-specific models as a query and then rank all resumes according to their similarity, again weighted cross-entropy.

3. EXPERIMENTS AND ANALYSIS

The resume/job matching experiments are performed on a challenging large scale real-world semi-structured collection. Each resume or job is represented as a record that may be missing some fields' information—i.e. some fields are NULL. Fields can be numeric or textual. In total, the collection contains 1,276,573 resume records (spanning 90 fields, 12 of them textual), 206,393 job records (spanning 20 fields, 9 of them textual) and 1,820,420 resume/job pairs annotated by implicit feedback from job agents. Table 1 shows some statistics for resume fields.

In experiments we first select a set of 300 jobs that had 60-80 annotated matching resumes. We split that set into two halves, one of which is used for training (e.g., tuning the Dirichlet smoothing parameters) and the other half is used for testing. In addition, we split the set of resumes equally into training and test sub-collections. We used the training resumes for building relevance models and searched for target resumes in the test resumes. When we incorporated structure for the SRM approach, we used the title and body fields from both resumes and jobs (even though they have the same name, the content is rarely similar).

Table 2 shows the performance of SRM against the two other approaches. We are matching 150 *test* jobs against the *test* resume collection. The upper half of Table 2 shows precision at fixed recall levels; the lower half shows precision at different ranks. The *%change* column shows relative difference between SRM and tRM. The *improved* column shows the number of matches where SRM exceeded tRM vs. the number of matches where performance was different. Bold figures indicate statistically significant differences (according to the sign test with $p < 0.05$).

The results show that a classic retrieval approach such as sLM performs very poorly for this task, suggesting that we cannot directly use text from job fields to find matching resumes. The Relevance Model approach achieves promising performance by incorporating a form of true relevance feedback. However, when structure is also provided, SRM outperforms tRM, beating tRM's mean average precision by almost 14%. R-precision and precision at 10 are improved by 17% and 19% respectively.

We note that performing this resume/job matching task

	sLM	tRM	SRM	%change	improved
Rel-ret:	242	1134	1255	**10.67**	**74/116**
Interpolated Recall - Precision:					
at 0.00	0.0299	0.2707	0.3133	**15.7**	**78/120**
at 0.10	0.0043	0.1547	0.1836	**18.6**	**72/100**
at 0.20	0.0019	0.1263	0.1439	**13.9**	**47/63**
at 0.30	0.0009	0.0839	0.0942	**12.2**	**25/35**
at 0.40	0.0003	0.0580	0.0625	**7.9**	**18/24**
Avg.Prec.	0.0018	0.0638	0.0726	**13.89**	**101/147**
Precision at:					
5 docs	0.0093	0.1627	0.1947	**19.7**	**23/33**
10 docs	0.0073	0.1460	0.1740	**19.2**	**31/41**
15 docs	0.0067	0.1289	0.1462	**13.4**	**34/51**
20 docs	0.0070	0.1113	0.1280	**15.0**	**40/58**
30 docs	0.0053	0.0876	0.1036	**18.3**	**48/61**
R-Prec.	0.0055	0.0824	0.0963	**16.95**	**52/68**

Table 2: Performance of matching 150 test jobs to the test resume collection. Evaluation is based on retrieving 1000 resumes. Across all 150 test jobs, there a total of 5173 matched resumes.

P@10	< 0.1	0.1-0.5	> 0.5
SRM	77	49	24
RM	87	45	18

Table 3: Counts broken down by P@10 ranges.

on a large-scale real-world semi-structured database is very difficult. At 5 documents retrieved, the precision of SRM is less than 20% while on average there are 35 annotated training resumes per job (half of the 60-80): that means that on average only 1 of the 35 relevant documents is found in the top five. To explore each *test* job's matching result further, we categorized the 150 jobs into 3 groups according to precision at 10; the size of each group is shown in Table 3. For some jobs, the relevance-based approaches find more than 5 matched resumes in the top 10 listed (i.e., P@10 is more than a half). By looking into the text of some failed matching cases directly we observe that judgments based on implicit feedback are still not good enough. Although still more analysis is needed, these preliminary results demonstrate that supervised feedback and structure are promising techniques for this difficult semi-structured documents matching task.

4. ACKNOWLEDGMENTS

We are indebted to Monster Worldwide's research lab for their continued support of this research. This work was also supported in part by the Center for Intelligent Information Retrieval and in part by the Defense Advanced Research Projects Agency (DARPA) under contract number HR0011-06-C-0023. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsors.

5. REFERENCES

[1] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of ACM SIGIR*, 2001.

[2] V. Lavrenko and W. B. Croft. Relevance-based language models. In *Proceedings of ACM SIGIR*, 2001.

[3] V. Lavrenko, X. Yi, and J. Allan. Information retrieval on empty fields. In *Proceedings of NAACL-HLT*, 2007.