Information Retrieval On Empty Fields

Victor Lavrenko, Xing Yi and James Allan
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, MA 01003-4610, USA
{lavrenko,yixing,allan}@cs.umass.edu

Abstract

We explore the problem of retrieving semi-structured documents from a real-world collection using a structured query. We formally develop Structured Relevance Models (SRM), a retrieval model that is 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. We then carry out a set of experiments using a snapshot of the National Science Digital Library (NSDL) repository, and queries that only mention fields missing from the test data. For such queries, typical field matching would retrieve no documents at all. In contrast, the SRM approach achieves a mean average precision of over twenty percent.

1 Introduction

This study investigates information retrieval on semi-structured information, where documents consist of several textual fields that can be queried independently. If documents contained *subject* and *author* fields, for example, we would expect to see queries looking for documents about *theory of relativity* by the author *Einstein*.

This setting suggests exploring the issue of inexact match—is *special theory of relativity* relevant?—that has been explored elsewhere (Cohen, 2000). Our interest is in an extreme case of that problem, where the content of a field is not corrupted or incorrect, but is actually absent. We wish to find relevant information in response to a query such as the one above even if a relevant document is completely missing the *subject* and *author* fields.

Our research is motivated by the challenges we encountered in working with the National Science Digital Library (NSDL) collection.[1] Each item in the collection is a scientific resource, such as a research paper, an educational video, or perhaps an entire website. In addition to its main content, each resource is annotated with *metadata*, which provides information such as the author or creator of the resource, its subject area, format (text/image/video) and intended audience – in all over 90 distinct fields (though some are very related). Making use of such extensive metadata in a digital library paves the way for constructing highly-focused models of the user's information need. These models have the potential to dramatically improve the user experience in targeted applications, such as the NSDL portals. To illustrate this point, suppose that we are running an educational portal targeted at elementary school teachers, and some user requests teaching aids for an introductory class on gravity. An intelligent search system would be able to translate the request into a structured query that might look something like: *subject='gravity' AND audience='grades 1-4' AND format='image,video' AND rights='free-for-academic-use'*. Such a query can be efficiently answered by a relational database system.

Unfortunately, using a relational engine to query a semi-structured collection similar to NSDL will run into a number of obstacles. The simplest problem is

[1] http://www.nsdl.org

Proceedings of NAACL HLT 2007, pages 89–96,
Rochester, NY, April 2007.

that natural language fields are filled inconsistently: e.g., the *audience* field contains values such as *K-4, K-6, second grade*, and *learner*, all of which are clearly semantically related.

A larger problem, and the one we focus on in this study, is that of missing fields. For example 24% of the items in the NSDL collection have no subject field, 30% are missing the author information, and over 96% mention no target audience (reading level). This means that a relational query for elementary school material will consider at most 4% of all potentially relevant resources in the NSDL collection.[2]

The goal of our work is to introduce a retrieval model that will be capable of answering complex structured queries over a semi-structured collection with corrupt and missing field values. This study focuses on the latter problem, an extreme version of the former. Our approach is to use a generative model to compute how plausible a word would appear in a record's empty field given the context provided by the other fields in the record.

The remainder of this paper is organized as follows. We survey previous attempts at handling semi-structured data in section 2. Section 3 will provide the details of our approach, starting with a high-level view, then providing a mathematical framework, and concluding with implementation details. Section 4 will present an extensive evaluation of our model on the large set of queries over the NSDL collection. We will summarize our results and suggest directions for future research in Section 5.

2 Related work

The issue of missing field values is addressed in a number of recent publications straddling the areas of relational databases and machine learning. In most cases, researchers introduce a statistical model for predicting the value of a missing attribute or relation, based on observed values. Friedman et al (1999) introduce a technique called Probabilistic Relational Models (PRM) for automatically learning the structure of dependencies in a relational database. Taskar et al (2001) demonstrate how PRM can be used to predict the category of a given research paper and show that categorization accuracy can be substantially improved by leveraging the relational structure of the data. Heckerman et al (2004) introduce the Probabilistic Entity Relationship (PER) model as an extension of PRM that treats relations between entities as objects. Neville at al (2003) discuss predicting binary labels in relational data using Relational Probabilistic Trees (RPT). Using this method they successfully predict whether a movie was a box office hit based on other movies that share some of the properties (actors, directors, producers) with the movie in question.

Our work differs from most of these approaches in that we work with free-text fields, whereas database researchers typically deal with closed-vocabulary values, which exhibit neither the synonymy nor the polysemy inherent in natural language expressions. In addition, the goal of our work is different: we aim for accurate *ranking* of records by their relevance to the user's query, whereas database research has typically focused on *predicting* the missing value.

Our work is related to a number of existing approaches to semi-structured text search. Desai et al (1987) followed by Macleod (1991) proposed using the standard relational approach to searching unstructured texts. The lack of an explicit ranking function in their approaches was partially addressed by Blair (1988). Fuhr (1993) proposed the use of Probabilistic Relational Algebra (PRA) over the weights of individual term matches. Vasanthukumar et al (1996) developed a relational implementation of the inference network retrieval model. A similar approach was taken by de Vries and Wilschut (1999), who managed to improve the efficiency of the approach. De Fazio et al (1995) integrated IR and RDBMS technology using an approached called cooperative indexing. Cohen (2000) describes WHIRL – a language that allows efficient inexact matching of textual fields within SQL statements. A number of relevant works are also published in the proceedings of the *INEX* workshop.[3]

The main difference between these endeavors and our work is that we are explicitly focusing on the cases where parts of the structured data are missing

[2] Some of the NSDL metadata fields overlap substantially in meaning, so it might be argued that the overlapping fields will cover the collection better. Under the broadest possible interpretation of field meanings, more than 7% of the documents still contain no subject and 95% still contain no audience field.

[3] http://inex.is.informatik.uni-duisburg.de/index.html

or mis-labeled.

3 Structured Relevance Model

In this section we will provide a detailed description of our approach to searching semi-structured data. Before diving into the details of our model, we want to clearly state the challenge we intend to address with our system.

3.1 Task: finding relevant records

The aim of our system is to identify a set of records relevant to a structured query provided by the user. We assume the query specifies a set of keywords for each field of interest to the user, for example *Q: subject='physics,gravity' AND audience='grades 1-4'*[4]. Each record in the database is a set of natural-language descriptions for each field. A record is considered relevant if it *could plausibly* be annotated with the query fields. For example, a record clearly aimed at elementary school students would be considered relevant to *Q* even if it does not contain *'grades 1-4'* in its description of the target audience. In fact, our experiments will specifically focus on finding relevant records that contain no direct match to the specified query fields, explicitly targeting the problem of missing data and inconsistent schemata.

This task is not a typical IR task because the fielded structure of the query is a critical aspect of the processing, not one that is largely ignored in favor of pure content based retrieval. On the other hand, the approach used is different from most DB work because cross-field dependencies are a key component of the technique. In addition, the task is unusual for both communities because it considers an unusual case where the fields in the query do not occur at all in the documents being searched.

3.2 Overview of the approach

Our approach is 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, a resource titled *'Transductive SVMs'* and containing highly technical language in its description is unlikely to be aimed at elementary-school students. In the following section we will describe a statistical model that will allow us to guess the values of un-observed fields. At the intuitive level, the model takes advantage of the fact that records similar in one respect will often be similar in others. For example, if two resources share the same author and have similar titles, they are likely to be aimed at the same audience. Formally, our model is based on the *generative* paradigm. We will describe a probabilistic process that could be viewed, hypothetically, as the source of every record in our collection. We will assume that the query provided by our user is also a sample from this generative process, albeit a very short one. We will use the observed query fields (e.g. *audience* and *subject*) to estimate the likely values for other fields, which would be *plausible* in the context of the observed subject and audience. The distributions over plausible values will be called *relevance models*, since they are intended to mimic the kind of record that might be relevant to the observed query. Finally, all records in the database will be ranked by their information-theoretic similarity to these relevance models.

3.3 Definitions

We start with a set of definitions that will be used through the remainder of this paper. Let C be a collection of semi-structured records. Each record $\mathbf{w}$ consists of a set of fields $\mathbf{w}_1 \ldots \mathbf{w}_m$. Each field $\mathbf{w}_i$ is a sequence of discrete variables (words) $\mathbf{w}_{i,1} \ldots \mathbf{w}_{i,n_i}$, taking values in the field vocabulary $\mathcal{V}_i$.[5] When a record contains no information for the i'th field, we assume n_i=0 for that record. A user's query $\mathbf{q}$ takes the same representation as a record in the database: $\mathbf{q}$=$\{\mathbf{q}_{i,j} \in \mathcal{V}_i : i$=$1..m, j = 1..n_i\}$. We will use $\mathbf{p}_i$ to denote a *language model* over $\mathcal{V}_i$, i.e. a set of probabilities $\mathbf{p}_i(v) \in [0,1]$, one for each word v, obeying the constraint $\Sigma_v \mathbf{p}_i(v) = 1$. The set of all possible language models over $\mathcal{V}_i$ will be denoted as the probability simplex $I\!\!P_i$. We define $\pi : I\!\!P_1 \times \cdots \times I\!\!P_m \rightarrow [0,1]$ to be a discrete measure function that assigns a probability mass $\pi(\mathbf{p}_1 \ldots \mathbf{p}_m)$ to a set of m language models, one for each of the m fields present in our collection.

[4] For this paper we will focus on simple conjunctive queries. Extending our model to more complex queries is reserved for future research.

[5] We allow each field to have its own vocabulary $\mathcal{V}_i$, since we generally do not expect author names to occur in the audience field, etc. We also allow $\mathcal{V}_i$ to share same words.

3.4 Generative Model

We will now present a generative process that will be viewed as a hypothetical source that produced every record in the collection C. We stress that this process is purely hypothetical; its only purpose is to model the kinds of dependencies that are necessary to achieve effective ranking of records in response to the user's query. We assume that each record $\mathbf{w}$ in the database is generated in the following manner:

1. Pick m distributions $\mathbf{p}_1 \ldots \mathbf{p}_m$ according to π
2. For each field $i = 1 \ldots m$:
   (a) Pick the length n_i of the $i'th$ field of $\mathbf{w}$
   (b) Draw i.i.d. words $\mathbf{w}_{i,1} \ldots \mathbf{w}_{i,n_i}$ from $\mathbf{p}_i$

Under this process, the probability of observing a record $\{\mathbf{w}_{i,j} : i{=}1..m, j{=}1..n_i\}$ is given by the following expression:

$$\int_{\mathbb{P}_1 \ldots \mathbb{P}_m} \left[\prod_{i=1}^{m} \prod_{j=1}^{n_i} \mathbf{p}_i(\mathbf{w}_{i,j}) \right] \pi(\mathbf{p}_1 \ldots \mathbf{p}_m) d\mathbf{p}_1 \ldots d\mathbf{p}_m \quad (1)$$

3.4.1 A generative measure function

The generative measure function π plays a critical part in equation (1): it specifies the likelihood of using different combinations of language models in the process of generating $\mathbf{w}$. We use a non-parametric estimate for π, which relies directly on the combinations of language models that are observed in the training part of the collection. Each training record $\mathbf{w}_1 \ldots \mathbf{w}_m$ corresponds to a unique combination of language models $\mathbf{p}_1^{\mathbf{w}} \ldots \mathbf{p}_m^{\mathbf{w}}$ defined by the following equation:

$$\mathbf{p}_i^{\mathbf{w}}(v) = \frac{\#(v, \mathbf{w}_i) + \mu_i c_v}{n_i + \mu_i} \quad (2)$$

Here $\#(v, \mathbf{w}_i)$ represents the number of times the word v was observed in the i'th field of $\mathbf{w}$, n_i is the length of the i'th field, and c_v is the relative frequency of v in the entire collection. Meta-parameters μ_i allow us to control the amount of smoothing applied to language models of different fields; their values are set empirically on a held-out portion of the data.

We define $\pi(\mathbf{p}_1 \ldots \mathbf{p}_m)$ to have mass $\frac{1}{N}$ when its argument $\mathbf{p}_1 \ldots \mathbf{p}_m$ corresponds to one of the N records $\mathbf{w}$ in the training part C_t of our collection, and zero otherwise:

$$\pi(\mathbf{p}_1 \ldots \mathbf{p}_m) = \frac{1}{N} \sum_{\mathbf{w} \in C_t} \prod_{i=1}^{m} 1_{\mathbf{p}_i = \mathbf{p}_i^{\mathbf{w}}} \quad (3)$$

Here $\mathbf{p}_i^{\mathbf{w}}$ is the language model associated with the training record $\mathbf{w}$ (equation 2), and 1_x is the Boolean indicator function that returns 1 when its predicate x is true and zero when it is false.

3.4.2 Assumptions and limitations of the model

The generative model described in the previous section treats each field in the record as a *bag* of words with no particular order. This representation is often associated with the assumption of *word independence*. We would like to stress that our model does not assume word independence, on the contrary, it allows for strong *un-ordered* dependencies among the words – both within a field, and across different fields within a record. To illustrate this point, suppose we let $\mu_i \rightarrow 0$ in equation (2) to reduce the effects of smoothing. Now consider the probability of observing the word *'elementary'* in the audience field together with the word *'differential'* in the title (equation 1). It is easy to verify that the probability will be non-zero only if some training record $\mathbf{w}$ actually contained these words in their respective fields – an unlikely event. On the other hand, the probability of *'elementary'* and *'differential'* co-occurring in the same title might be considerably higher.

While our model does not assume word independence, it does ignore the relative ordering of the words in each field. Consequently, the model will fail whenever the order of words, or their proximity within a field carries a semantic meaning. Finally, our generative model does not capture dependencies across different records in the collection, each record is drawn independently according to equation (1).

3.5 Using the model for retrieval

In this section we will describe how the generative model described above can be used to find database records relevant to the structured query provided by the user. We are given a structured query $\mathbf{q}$, and a collection of records, partitioned into the training portion C_t and the testing portion C_e. We will use the training records to estimate a set of *relevance*

	records covered	average length	unique words
title	655,673 (99%)	7	102,772
description	514,092 (78%)	38	189,136
subject	504,054 (77%)	12	37,385
content	91,779 (14%)	743	575,958
audience	22,963 (3.5%)	4	119

Table 1: Summary statistics for the five NSDL fields used in our retrieval experiments.

models $R_1 ... R_m$, intended to reflect the user's information need. We will then rank testing records by their divergence from these relevance models. A relevance $R_i(v)$ specifies how plausible it is that word v would occur in the i'th field of a record, given that the record contains a perfect match to the query fields $\mathbf{q}_1 ... \mathbf{q}_m$:

$$R_i(v) = \frac{P(\mathbf{q}_1 ... v \circ \mathbf{q}_i ... \mathbf{q}_m)}{P(\mathbf{q}_1 ... \mathbf{q}_i ... \mathbf{q}_m)} \quad (4)$$

We use $v \circ \mathbf{q}_i$ to denote appending word v to the string $\mathbf{q}_i$. Both the numerator and the denominator are computed using equation (1). Once we have computed relevance models R_i for each of the m fields, we can rank testing records $\mathbf{w}'$ by their similarity to these relevance models. As a similarity measure we use weighted cross-entropy, which is an extension of the ranking formula originally proposed by (Lafferty and Zhai, 2001):

$$H(R_{1..m}; \mathbf{w}_{1..m}) = \sum_{i=1}^{m} \alpha_i \sum_{v \in \mathcal{V}_i} R_i(v) \log \mathbf{p}_i^{\mathbf{w}}(v) \quad (5)$$

The outer summation goes over every field of interest, while the inner extends over all the words in the vocabulary of the i'th field. R_i are computed according to equation (4), while $\mathbf{p}_i^{\mathbf{w}}$ are estimated from equation (2). Meta-parameters α_i allow us to vary the importance of different fields in the final ranking; the values are selected on a held-out portion of the data.

4 Experiments

4.1 Dataset and queries

We tested the performance of our model on a January 2005 snapshot of the National Science Digital Library repository. The snapshot contains a total of 656,992 records, spanning 92 distinct (though sometimes related) fields. [6]Only 7 of these fields are present in every record, and half the fields are present in less than 1% of the records. An average record contains only 17 of the 92 fields. Our experiments focus on a subset of 5 fields (*title*, *description*, *subject*, *content* and *audience*). These fields were selected for two reasons: (i) they occur frequently enough to allow a meaningful evaluation and (ii) they seem plausible to be included in a potential query.[7] Of these fields, *title* represents the title of the resource, *description* is a very brief abstract, *content* is a more detailed description (but not the full content) of the resource, *subject* is a library-like classification of the topic covered by the resource, and *audience* reflects the target reading level (e.g. *elementary school* or *post-graduate*). Summary statistics for these fields are provided in Table 1.

The dataset was randomly split into three subsets: the **training** set, which comprised 50% of the records and was used for estimating the relevance models as described in section 3.5; the **held-out** set, which comprised 25% of the data and was used to tune the smoothing parameters μ_i and the bandwidth parameters α_i; and the **evaluation** set, which contained 25% of the records and was used to evaluate the performance of the tuned model[8].

Our experiments are based on a set of 127 automatically generated queries. We randomly split the queries into two groups, 64 for training and 63 for evaluation. The queries were constructed by combining two randomly picked *subject* words with two *audience* words, and then discarding any combination that had less than 10 exact matches in any of the three subsets of our collection. This procedure yields queries such as Q_{91}={*subject:'artificial intelligence' AND audience='researchers'*}, or Q_{101}={*subject:'philosophy' AND audience='high school'*}.

4.2 Evaluation paradigm

We evaluate our model by its ability to find "relevant" records in the face of missing values. We de-

[6] As of May 2006, the NSDL contains over 1.5 million documents.

[7] The most frequent NSDL fields (*id, icon, url, link* and 4 *brand* fields) seem unlikely to be used in user queries.

[8] In real use, typical pseudo relevance feedback scheme can be followed: retrieve top-k documents to build relevance models then perform IR again on the same whole collection

fine a record $\mathbf{w}$ to be relevant to the user's query $\mathbf{q}$ if every keyword in $\mathbf{q}$ is found in the corresponding field of $\mathbf{w}$. For example, in order to be relevant to Q_{101} a record must contain the word *'philosophy'* in the subject field and words *'high'* and *'school'* in the audience field. If either of the keywords is missing, the record is considered non-relevant.[9]

When the testing records are fully observable, achieving perfect retrieval accuracy is trivial: we simply return all records that match all query keywords in the subject and audience fields. As we stated earlier, our main interest concerns the scenario when parts of the testing data are missing. We are going to simulate this scenario in a rather extreme manner by *completely* removing the *subject* and *audience* fields from all testing records. This means that a straightforward approach – matching query fields against record fields – will yield no relevant results. Our approach will rank testing records by comparing their *title, description* and *content* fields against the query-based relevance models, as discussed in section 3.5.

We will use the standard rank-based evaluation metrics: *precision* and *recall*. Let N_R be the total number of records relevant to a given query, suppose that the first K records in our ranking contain N_K relevant ones. Precision at rank K is defined as $\frac{N_K}{K}$ and recall is defined as $\frac{N_K}{N_R}$. Average precision is defined as the mean precision over all ranks where relevant items occur. R-precision is defined as precision at rank $K{=}N_R$.

4.3 Baseline systems

Our experiments will compare the ranking performance of the following retrieval systems:

cLM is a *cheating* version of un-structured text search using a state-of-the-art language-modeling approach (Ponte and Croft, 1998). We disregard the structure, take all query keywords and run them against a *concatenation* of all fields in the testing records. This is a "cheating" baseline, since the concatenation includes the *audience* and *subject* fields, which are supposed to be missing from the testing records. We use Dirichlet smoothing (Lafferty and Zhai, 2001), with parameters optimized on the training data. This baseline mimics the core search capability currently available on the NSDL website.

bLM is a combination of SQL-like structured matching and unstructured search with query expansion. We take all training records that contain an exact match to our query and select 10 highly-weighted words from the *title*, *description*, and *content* fields of these records. We run the resulting 30 words as a language modeling query against the concatenation of *title*, *description*, and *content* fields in the testing records. This is a non-cheating baseline.

bMatch is a structured extension of bLM. As in bLM, we pick training records that contain an exact match to the query fields. Then we match 10 highly-weighted *title* words, against the *title* field of testing records, do the same for the *description* and *content* fields, and merge the three resulting ranked lists. This is a non-cheating baseline that is similar to our model (SRM). The main difference is that this approach uses exact matching to select the training records, whereas SRM leverages a best-match language modeling algorithm.

SRM is the Structured Relevance Model, as described in section 3.5. For reasons of both effectiveness and efficiency, we firstly run the original query to retrieve top-500 records, then use these records to build SRMs. When calculating the cross entropy(equ. 5), for each field we only include the top-100 words which will appear in that field with the largest probabilities.

Note that our baselines do not include a standard SQL approach directly on testing records. Such an approach would have perfect performance in a "cheating" scenario with observable *subject* and *audience* fields, but would not match any records when the fields are removed.

4.4 Experimental results

Table 2 shows the performance of our model (SRM) against the three baselines. The model parameters were tuned using the 64 training queries on the *training* and *held-out* sets. The results are for the 63 test queries run against the *evaluation* corpus. (Similar results occur if the 64 training queries are run against

[9] This definition of relevance is unduly conservative by the standards of Information Retrieval researchers. Many records that might be considered relevant by a human annotator will be treated as non-relevant, artificially decreasing the accuracy of any retrieval algorithm. However, our approach has the advantage of being fully automatic: it allows us to test our model on a scale that would be prohibitively expensive with manual relevance judgments.

	cLM	bMatch	bLM	SRM	%change	improved
Rel-ret:	949	582	914	861	-5.80	26/50
Interpolated Recall - Precision:						
at 0.00	0.3852	0.3730	0.4153	0.5448	31.2	**33/49**
at 0.10	0.3014	0.3020	0.3314	0.4783	44.3	**42/56**
at 0.20	0.2307	0.2256	0.2660	0.3641	36.9	**40/59**
at 0.30	0.2105	0.1471	0.2126	0.2971	39.8	**36/58**
at 0.40	0.1880	0.1130	0.1783	0.2352	31.9	**36/58**
at 0.50	0.1803	0.0679	0.1591	0.1911	20.1	32/57
at 0.60	0.1637	0.0371	0.1242	0.1439	15.8	27/51
at 0.70	0.1513	0.0161	0.1001	0.1089	8.7	21/42
at 0.80	0.1432	0.0095	0.0901	0.0747	-17.0	18/36
at 0.90	0.1292	0.0055	0.0675	0.0518	-23.2	12/27
at 1.00	0.1154	0.0043	0.0593	0.0420	-29.2	9/23
Avg.Prec.	0.1790	0.1050	0.1668	0.2156	29.25	**43/63**
Precision at:						
5 docs	0.1651	0.2159	0.2413	0.3556	47.4	**32/43**
10 docs	0.1571	0.1651	0.2063	0.2889	40.0	**34/48**
15 docs	0.1577	0.1471	0.1841	0.2360	28.2	**32/49**
20 docs	0.1540	0.1349	0.1722	0.2024	17.5	28/47
30 docs	0.1450	0.1101	0.1492	0.1677	12.4	29/50
100 docs	0.0913	0.0465	0.0849	0.0871	2.6	**37/57**
200 docs	0.0552	0.0279	0.0539	0.0506	-6.2	33/53
500 docs	0.0264	0.0163	0.0255	0.0243	-4.5	26/48
1000 docs	0.0151	0.0092	0.0145	0.0137	-5.8	26/50
R-Prec.	0.1587	0.1204	0.1681	0.2344	39.44	**31/49**

Table 2: Performance of the 63 test queries retrieving 1000 documents on the evaluation data. Bold figures show statistically significant differences. Across all 63 queries, there are 1253 relevant documents.

the *evalution* corpus.)

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 our model and the baseline bLM. The *improved* column shows the number of queries where SRM exceeded bLM vs. the number of queries where performance was different. For example, $33/49$ means that SRM out-performed bLM on 33 queries out of 63, underperformed on $49-33=16$ queries, and had exactly the same performance on $63-49=14$ queries. Bold figures indicate statistically significant differences (according to the sign test with $p < 0.05$).

The results show that SRM outperforms three baselines in the high-precision region, beating bLM's mean average precision by 29%. User-oriented metrics, such as R-precision and precision at 10 documents, are improved by 39.4% and 44.3% respectively. The absolute performance figures are also very encouraging. Precision of 28% at rank 10 means that on average almost 3 out of the top 10 records in the ranked list are relevant, despite the requested fields not being available to the model.

We note that SRM continues to outperform bLM until very high recall and until the 100-document cutoff. After that, SRM degrades rapidly with respect to bLM. We feel the drop in effectiveness is of marginal interest because precision is already well below 10% and few users will be continuing to that depth in the list.

It is encouraging to see that SRM outperforms both cLM, the cheating baseline that takes advantage of the field values that are supposed to be "missing", and bMatch, suggesting that best-match retrieval provides a superior strategy for selecting a set of appropriate training records.

5 Conclusions

We have developed and empirically validated a new retrieval model for semi-structured text. The model is based on the idea that missing or corrupted values for one field can be inferred from values in other fields of the record. The cross-field inference makes it possible to find documents in response to a structured query when those query fields do not exist in the relevant documents at all.

We validated the SRM approach on a large archive of the NSDL repository. We developed a large set of structured Boolean queries that had relevant documents in the test portion of collection. We then indexed the documents *without* the fields used in the queries. As a result, using standard field matching approaches, not a single document would be returned in response to the queries—in particular, no relevant documents would be found.

We showed that standard information retrieval techniques and structured field matching could be combined to address this problem, but that the SRM approach outperforms them. We note that SRM brought two relevant documents into the top five—again, querying on missing fields—and achieved an average precision of 23%, a more than 35% improvement over a state-of-the-art relevance model approach combining the standard field matching.

Our work is continuing by exploring methods for handling fields with incorrect or corrupted values. The challenge becomes more than just inferring what values might be there; it requires combining likely missing values with confidence in the values already present: if an audience field contains 'undergraduate', it should be unlikely that 'K-6' would be a plausible value, too.

In addition to using SRMs for retrieval, we are currently extending the ideas to provide field validation and suggestions for data entry and validation: the same ideas used to find documents with missing field values can also be used to suggest potential values for a field and to identify values that seem inappropriate. We have also begun explorations toward using inferred values to help a user browse when starting from some structured information—e.g., given values for two fields, what values are probable for other fields.

Acknowledgments

This work was 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 sponsor.

References

D.C. Blair. 1988. An extended relational document retrieval model. *Inf. Process. Manage.*, 24(3):349–371.

W.W. Cohen. 2000. WHIRL: A word-based information representation language. *Artificial Intelligence*, 118(1–2):163–196.

S. DeFazio, A. Daoud, L. A. Smith, and J. Srinivasan. 1995. Integrating IR and RDBMS Using Cooperative Indexing. In *Proceedings of SIGIR*, pages 84–92.

B. C. Desai, P. Goyal, and F. Sadri. 1987. Non-first normal form universal relations: an application to information retrieval systems. *Inf. Syst.*, 12(1):49–55.

N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. 1999. Learning probabilistic relational models. In *IJCAI*, pages 1300–1309.

N. Fuhr. 1993. A probabilistic relational model for the integration of IR and databases. In *Proceedings of SIGIR*, pages 309–317.

D. Heckerman, C. Meek, and D. Koller. 2004. Probabilistic models for relational data. Technical Report MSR-TR-2004-30, Microsoft Research.

J. Lafferty and C. Zhai. 2001. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of SIGIR*, pages 111–119.

I. Macleod. 1991. Text retrieval and the relational model. *Journal of the American Society for Information Science*, 42(3):155–165.

J. Neville, D. Jensen, L. Friedland, and M. Hay. 2003. Learning relational probability trees. In *Proceedings of ACM KDD*, pages 625–630, New York, NY, USA.

J. M. Ponte and W. B. Croft. 1998. A language modeling approach to information retrieval. In *Proceedings of SIGIR*, pages 275–281.

B. Taskar, E. Segal, and D. Koller. 2001. Probabilistic classification and clustering in relational data. In *Proceedings of IJCAI*, pages 870–876.

S. R. Vasanthakumar, J.P. Callan, and W.B. Croft. 1996. Integrating INQUERY with an RDBMS to support text retrieval. *IEEE Data Eng. Bull.*, 19(1):24–33.

A.D. Vries and A. Wilschut. 1999. On the integration of IR and databases. In *Proceedings of IFIP 2.6 Working Conf. on Data Semantics*, Rotorua, New Zealand.