Discovering Missing Values in Semi-Structured Databases

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

Abstract

We explore the problem of discovering multiple missing values in a semi-structured database. For this task, we formally develop Structured Relevance Model (SRM) built on one hypothetical generative model for semi-structured records. SRM 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. Small-scale experiments on IMDb (Internet Movie Database) show that SRM matched three state-of-the-art relational learning approaches on the movie label prediction tasks. Large-scale experiments on a snapshot of the National Science Digital Library (NSDL) repository show that SRM is highly effective at discovering possible values for free-text fields even with quite modest amounts of training data, compared with state-of-the-art machine learning approaches.

1. Introduction

Recently, information processing challenges on semi-structured data have attracted more and more researchers. One approach arises out of the relational database community and uses SQL with a typical relational query (Grabs & Schek 2002). Such an SQL approach usually assumes complete information for every record in the database, which means it runs into obstacles when dealing with real-world semi-structured data, where some field values or even whole fields information are missing. For example, a query: *subject* = '*elementary differential*' AND *audience* = '*undergraduate*' might miss many plausible relevant records about "elementary differential" only because they lack the target audience (reading level) information.

One intuitive way to solve this problem is to divide the retrieval process into two steps: first discover the missing values for a given field by using context information provided by other fields in the record, and second, perform typical retrieval on all the records with this reconstructed complete information. In this paper, we will focus on the first step, inferring missing field values. We will formally develop Structured Relevance Model (SRM), a generative approach to address the problem.

This research is motivated by the challenges we encountered in working with the National Science Digital Library (NSDL) collection.[1] In this collection each record 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). Table 1 shows some statistics of 5 fields (*title*, *description*, *subject*, *content* and *audience*) from a January 2004 snapshot of the NSDL collection. It can be observed that 23% of

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

the records in the collection have no *subject* field and only 3.5% mention target audience. Therefore if a relational engine were directly applied for querying records in the NSDL collection, it will bump into the missing field problem. For example if a query contains "*audience = elementary school*", it will consider at most 3.5% of all potentially relevant resources in the NSDL collection. However if the missing field values can be plausibly inferred according to the other observed field values in the record, this coverage problem will be alleviated and additional plausible relevant records in the collection will be retrieved.

	records covered	average length	unique values
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 five NSDL fields.

Discovering missing field values on collections similar to NSDL can be viewed as a missing label problem in the multi-label machine learning scenario. But if we are directly applying a multi-label learning algorithm to predict missing field values, there is a big obstacle: fields usually consist of free text values instead of closed-vocabulary small size labels. Assuming that each unique value in a field is a label for a different category, it can be observed in Table 1 that there are 119 categories in the multi-label learning task for the *audience* field, and 37,385 categories for the *subject* field. To the best of our knowledge, there are few multi-label machine learning algorithms that directly address this missing label prediction problem with huge label variety. In this paper, for this challenging task, we present the SRM approach, which is built on one generative model for semi-structured records and follows the language model approach that has been successfully applied to retrieval tasks. Intuitively SRM is able to calculate the probability of a value occurring in certain fields of a record, then utilizes the rank list of these probabilities to discover some import missing field values.

Our experiments show that SRM's performance is comparable with state-of-the-art machine learning approaches. For the missing field discovery task on the NSDL collection, it is found that SRM is even preferable to Support Vector Machine (SVM) in the learning scenario where there are huge number of categories and modest number of training samples for each category. Therefore SRM can be easily adopted for the final task of answering complex structured queries over a semi-structured collection with corrupt and missing field values.

The remainder of the paper is organized as follows. We survey previous attempts at handling semi-structured data and predicting missing values in section 2. Section 3 will provide the details of Structured Relevance Models. In section 4, we will discuss the relation between SRM and state-of-the-art relational learning approaches, and then compare their performance on the movie receipts predicting task with the IMDb collection. In section 5, the challenging NSDL collection, where numerous free-text field values are missing, will be utilized to show how SRM performs and why SRM is preferable than some state-of-the-art machine learning approach for the missing value discovery task on it. Section 6 will discuss the results and draw conclusions.

2. Related Work

Most of the prior work related to our research for discovering missing values in relational databases can be grouped into two categories: (i) research on dealing with missing data in relational databases, and (ii) research on dealing with multi-label data in machine learning task. The issue of discovering missing field values in relational databases has recently been addressed in a few publications in relational machine learning area. In the literature, researchers usually introduce a statistical model for predicting the value of a missing attribute or relation, based on observed values. Friedman *et al.* (1999) introduced a directed graphical model, Probabilistic Relational Models (PRM) that extends Bayesian networks for automatically reasoning in a relational database. Taskar *et al.* (2001) demonstrated how PRM can be used to predict the category of a given research paper and showed that categorization accuracy can be substantially improved by leveraging the relational structure of the data. They also explored undirected graphical models capable of representing and reasoning with autocorrelation in relational data and introduced relational Markov networks (RMNs) (Taskar *et al.*, 2002), based on conditional random fields for sequence data (Lafferty *et al.* 2001). Neville *et al.* (2003*a*, 2003*b*, 2003*c*) discussed predicting binary labels in relational data using Relational Naive Bayesian Classifier (RBC), Relational Probabilistic Trees (RPT) and Relational Dependency Networks (RDN). Using these powerful relational learning models they successfully predicted 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 relational learning researchers typically deal with closed-vocabulary values, which exhibit neither the synonymy nor the polysemy inherent in natural language expressions. Furthermore, usually there are thousands of different missing values needed to be addressed in this NSDL collection, whereas typical relational learning task focused on predicting and selecting the missing labels for a given record from several given labels or categories.

Our work is also related to machine learning research on multi-labeled classification problems. Multi-labeled classification requires classifiers to assign each sample into more than one category, which brings complexity and difficulty to the learning task. Zhu *et al.* (2005) provide a detailed survey for different approaches of multi-labeled classification techniques and more recent work is presented by Rousu *et al.* (2006). Instead of building complicated hierarchical learning models (Godbole & Sarawagi, 2004; Rousu *et al.*, 2006), our research follows two other simple but important approaches: a ranking-based approach and a generative probabilistic model based approach. The ranking-based approach usually computes some real-value score for each sample-category pair, and classifies each sample by choosing all the categories having the scores above the given threshold. Many classic machine learning techniques have been adopted for learning the ranking function for the sample-category pairs, for example Schapire and Singer (2000) modified boosting techniques for this task, whereas Elisseeff and Weston (2002) utilized Support Vector Machines (SVMs). The generative approach usually creates generative probabilistic models for the process of generating the multi-labeled samples, and learns posteriors for classification. McCallum (1999) described one parametric generative mixture model which assumes that each multi-labeled sample is generated by a mixture of single-labeled generative models, then utilized EM algorithm for learning parameters; Ueda and Saito (2003) proposed two new different mixture schemes into the generative models and achieved a little bit better performance than utilizing single-labeled learning algorithms for multi-labeled task.

Our work differs from previous work following these two approaches in that we focus on the specific task of discovering missing values in semi-structured database. We also directly handle large scale incomplete semi-structured data where there are a great number of missing fields. In addition, we borrow ideas from language model based relevance models (Ponte & Croft, 1998; Lavrenko & Croft, 2001) and follow the non-parametric approach (Silverman, 1986) to build the generative function for our hypothetical generative process of creating semi-structured records. For the semi-structured document classification problem, there are some recent approaches in machine learning committee by firstly building generative models then doing the classifications (Diligenti *et al.*, 2001; Denoyer & Gallinari, 2004). Different from these approaches, we focus on the specific NSDL task where there are a great number of missing labels.

3. Structured Relevance Models

In this section, we will firstly describe the structured generative model for Structured Relevance Model (SRM) in detail, then discuss its assumption and limitation and finally describe how to utilize this generative model to build SRM and discover missing values in semi-structured database.

3.1 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 $\mathcal{V}_i$.[2] When a record contains no information for the $i'th$ field, we assume $n_i = 0$ for that record. 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 $\sum_v \mathbf{p}_i(v) = 1$. The set of all possible language models over $\mathcal{V}_i$ will be denoted as the probability simplex $\mathbb{P}_i$. We define $\pi : \mathbb{P}_1 \times \cdots \mathbb{P}_m \rightarrow [0,1]$ 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.

3.2 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 among words within and between fields so that they can be utilized to discover the missing field values given the context provided by other field values in the record.

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 $\mathbf{p}_i$

[2] 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.

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

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

A Generative Measure function

The generative measure function π plays a critical part: it specifies the likelihood of using different combinations of language models in the process of generating $\mathbf{w}$. The measure function can be set in a number of different ways, leading to very different dependence structures among the fields of $\mathbf{w}$. In choosing π we tried to make as few assumptions as possible about the structure of our collection, allowing the data to speak for itself. We use a non-parametric estimate for π, which makes our generative model similar to *Parzen windows* or *kernel-based* density estimators (Silverman, 1986). Our estimate 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:

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

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 $1/N$ 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:

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

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 is true and zero when it is false.

Assumptions and Limitations

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 $\mu_i \to 0$ 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).

3.3 Structured Relevance Models for Discovering Missing Values

In this section we will describe how the generative model described above can be used to build SRMs for discovering missing field values in semi-structured database.

Suppose that the whole collection has been divided into training set C_t and testing set C_e. We are given a structured record $\mathbf{r}$ from C_e, which has observed field values $\mathbf{r}_1 \ldots \mathbf{r}_m$ and also missed some field values. Now we estimate the likely missing values which would be *plausible* in the context of the observed parts by using C_t. The distribution over all plausible values in the *i'th* field is called *relevance model* R_i for the *i'th* field since it is intended to mimic the field values that might be relevant to the observed record $\mathbf{r}$. A set of relevance models $R_1 \ldots R_m$ for all the fields will be estimated, which are called Structured Relevance Models (SRMs). Formally speaking, a relevance model $R_i(v)$ in a SRMs specifies how plausible it is that word v would occur in the *i'th* field of a record, given that the observed parts of the record are $\mathbf{r}_1 \ldots \mathbf{r}_m$:

$$R_i(v) = \frac{P(\mathbf{r}_1 \ldots v \circ \mathbf{r}_i \ldots \mathbf{r}_m)}{P(\mathbf{r}_1 \ldots \mathbf{r}_i \ldots \mathbf{r}_m)} \tag{4}$$

Here we use $v \circ \mathbf{r}_i$ to denote appending word v to the string $\mathbf{r}_i$. Both the numerator and denominator are computed using equation (1) based on the described generative model. Once we have computed a SRM for each field, we can rank plausible values in that field according to their probabilities in the corresponding relevance model. By keeping the top-*k* values in this rank list, we can discover the important missing field values for record $\mathbf{r}$.

3.4 Implementation details

In this section, we will briefly describe how to the model has been implemented in practice. We will use the following example which closely resembles the experiments we carried out on the NSDL collection.

Suppose that the NSDL collection contains these five fields: *title, description, content, subject, audience*. Assume that all five fields are observable in the training records C_t, and that *audience* and *subject* are missing in **every** testing record in C_e. We proceed as follows to discover the missing field values of testing records:

1. Given a record $\mathbf{r} = \{\mathbf{r}_t, \mathbf{r}_d, \mathbf{r}_c\}$ from testing records C_e. Run $\mathbf{r}_t$ as a query against the *title* index on the training records C_t. Run $\mathbf{r}_d$ against *description* index and $\mathbf{r}_c$ against *content* index.
2. Merge the ranked lists from the *title*, *description* and *content* queries. The final score of record $\mathbf{w}$ in the merged list should be $P(\mathbf{r} \mid \mathbf{w}) = P(\mathbf{r}_t \mid \mathbf{w}_t) \times P(\mathbf{r}_d \mid \mathbf{w}_d) \times P(\mathbf{r}_c \mid \mathbf{w}_c)$
3. Take the top 1000 records and convert their scores to posterior probabilities $P(\mathbf{w} \mid \mathbf{r})$
4. Estimate a *audience* relevance model R_a by averaging relative frequencies of audience words in the top 1000 records, weighing each record by $P(\mathbf{w} \mid \mathbf{r})$. Repeat that for the *subject* field to estimate *subject* relevance model R_s.

5. Rank *audience* values according to R_a, keep the top-*k* field values as $\mathbf{r}_a$. Similarly, get $\mathbf{r}_s$ by using R_s. Finally return $\mathbf{r}_a$ and $\mathbf{r}_s$ as *plausible* missing values in *audience* and *subject* fields of $\mathbf{r}$.

The procedure utilizes a text search engine that ranks documents using the standard language-modeling approach (Ponte & Croft, 1998) and returns $P(\mathbf{w}|\mathbf{r})$ in the ranked list. The computational complexity of this procedure is depended on the number of values in each field of $\mathbf{r}$: $\{n_t, n_d, n_c\}$ in this example, and the number of field values in returned top-1000 records in step 3. In order to calculate $P(\mathbf{r}|\mathbf{w})$ by using typical text search engine, we can utilize $O(n_t + n_d + n_c)$ inverted document lists. In order to calculate relevance models for each field R_i, we need to compute the scores for each value in the returned top-1000 records and sort them: denote n'_i as the average length of the *i*th field of a record, N_i as the number of distinct values in the *i*th field of top-1000 records returned, the computation cost of this procedure is $\sum_i O(1000n'_i + N_i \log N_i)$. Therefore, although the whole procedure is a little computational expensive, we can still implement it efficiently with a typical text search engine.

4. SRM and Relational Learning Approach

SRM can be utilized for a typical classification task in a semi-structured database by viewing the *observed* field values as features and the *missing* ones as labels to be predicted. Intuitively, SRM utilizes the fact that records similar in one respect will often be similar in others, which is similar to the *autocorrelation* property (i.e., the same type of objects that are related to each other usually have similar attributes—e.g., movies produced by the same director may have similar genre) in relational data (Jensen & Neville, 2002). Therefore SRM is closely related to relational learning approaches, which deal directly with relational data and utilize relational features and attributes of related objects in the training process for the final classification.

In this section, we will utilize IMDb (the Internet Movie Database) to explore SRM's capability of predicting missing labels. We borrowed a classification task Neville *et al.* (2003*b*; 2003*c*) where the goal is to predict whether a movie's opening weekend receipts will be larger than US$2 million or not, and compare SRM's performance with different relational learning approaches.

4.1 Experiments with IMDb

This experiment utilized movie data drawn from the Internet Movie Database (IMDb).[3] A sample of all movies released in the United States from 1996 through 2000 is gathered, with opening weekend receipt information. The resulting collection contains 1,362 movies. In addition to movies, the data set contains associated actors, directors, producers, and studios. The learning task was to predict whether a movie's opening-weekend box office receipts are larger than US$2 million or not (in this data set, 45% of the movies have that property). We compare our results with three state-of-the-art approaches in relational learning:

1. Relational Bayesian classifier (RBC) is a modification of the typical Naïve Bayesian classifier (Neville *et al.*, 2003*a*)
2. Relational Probabilistic Tree (RPT) is a modification of the typical probabilistic classification tree, which uses a novel form of randomization test to adjust for statistical bias in relational data (Neville *et al.*, 2003*b*)

[3] www.imdb.com

3. Relational Dependency Networks (RDN) is an undirected graphical model which is capable of reasoning with autocorrelation (Neville *et al.*, 2003*c*). RDN utilizes Gibbs sampling and RPT for learning parameters.

For using these relational learning models[4] for a movie's receipt label prediction, we follow the same setting as Neville *et al.* (2003*c*) and utilize both attributes of a given movie and the attributes of objects one or two links away from this given movie – directors, actors and studios, as well as movies associated with these objects.

For using SRM, we created a semi-structured record for each movie with the movie's attributes as well as all attributes from only adjacent directors, actors and studios; if there are multiple instances of one of those records, we concatenated them, e.g. all actor name fields would be combined into a single text field containing all the names. In this way we flattened the heterogeneous data into homogeneous records for SRM, which is similar in the preprocessing step in RBC (Neville *et al.*, 2003*a*). Table 2 shows statistics of some important attributes of this movie data set after this flattening step. Note that all the numeric values have been discretized such as actor's birth year and a person's "Hollywood Stock Exchange" (HSX) rating[5].

	records covered	unique values	avg length
actor_HSX_rating	1035	10	4
actor_birth_year	1305	10	12
actor_has_award_or_nominated	1354	2	40
actor_gender	1354	2	40
actor_name	1354	37988	40
movie_is_comedy	1362	2	1
movie_is_drama	1362	2	1
movie_genre	1362	17	2
first_movie_year	1360	10	5
studio_in_USA	1360	2	5
studio_name	1360	748	5
first_year_directed	1360	7	1
director_has_award_or_nominated	1360	2	1
director_name	1360	1126	1

Table 2. Summary statistics for some important attributes in this movie samples.

Table 2 shows many detailed characteristics of this movie data set, e.g. the average length of *actor_name* field being 40 means that in average there are 40 actors for each movie. It can be observed that in this movie data set, most fields are not missing in records: the field covered by the fewest records is *actor_HSX_rating*, which appears in 1,035 records. It should be pointed out that many of these attributes have been utilized in experiments reported by Neville *et al.* (2003*c*), although not exactly the same set.

[4] The Java versions of these relational learning models are included in the PROXIMITY project, downloadable at http://kdl.cs.umass.edu/proximity/.

[5] Hollywood Stock Exchange is a web-based, multiplayer game in which players use simulated money to buy and sell "shares" of actors, directors, upcoming films, and film-related options. Check http://en.wikipedia.org/wiki/-Hollywood_Stock_Exchange for more details.

	RBC	RPT	RDN	SRM
1	0.801	0.688	0.719	0.734
2	0.801	0.812	0.841	0.812
3	0.79	0.806	0.806	0.799
4	0.796	0.818	0.829	0.836
avg	0.797	0.781	0.799	0.795
stdev	0.005	0.062	0.055	0.043

Table 3. Accuracy results of IMDb task.

In experiments, this movie set (1,362 movies) is temporally split into five sets, one for each year from 1996 to 2000, and we carry out four experiments: test on the movies from one year with training data taken from all previous years. Similar to the experiments by Neville *et al.* (2003*c*), links to the future were removed from each testing sample; in contrast to their approach, links to the future were kept in each training sample for learning RBC, RPT and RDN. This is intended to reproduce the expected domain of application for these models.

Table 3 shows the accuracy of different approaches for this learning task. On this binary label prediction task with one small scale semi-structured dataset which has only a few missing fields, SRM matched the state-of-the-art relational learning approaches.

It is worth pointing out that in the data flattening step for SRM we do not include attributes of objects two links away from a given movie, which is different from the training data used for RBC, RPT and RDN. However SRM can implicitly use those related movies' information by knowing that they have similar director, actor, or studio field values. Therefore SRM can still achieve comparable performance with these powerful relational learning models.

4.2 Relations and Difference

There are many interesting relations between the SRM approach and the relational learning approach which can be found by looking at them closely. Similar to RBC, currently SRM can only deal with semi-structured data already flattened. Different from RBC, SRM follows a non-parametric approach for representing the probabilistic density function, which can employ a rich family of kernels in the generative function – equation (3) is only one choice for the generative function where the Dirac δ kernel function has been utilized. Different from relational learning approaches, SRM does not capture dependencies across different records in the collection and assume each record is drawn i.i.d. On the other hand, SRM represents the fact that records similar in one respect will often be similar in others by using one hypothetical generative models described in section 3.2 and successfully uses it for learning. We should admit that, because SRM discards the relational information in the original data – similar to RBC approach – it lacks the interpretability of the selective models generated by RPT and RDN.

5. SRMs for Discovering Multiple Missing Values in NSDL

In order to explore SRM's inference capability further, we return to our large scale semi-structured database, the NSDL collection, and compare SRM's performance on it with several other typical machine learning approaches. The goal here is to determine the values that should be in the *audience* and *subject* fields.

First we confine ourselves to a subset of the NSDL collection described in Table 4, which has all five of the *title*, *content*, *description*, *subject* and *audience* fields, and that has exactly one of *high school*, *middle school*, *undergraduate*, *graduate+research,* or *elementary school* in the *audience* field.

Audience Value	Num of Samples
High school	5210
Middle school	1520
Undergraduate	91
Graduate or Research	69
Elementary school	536

Table 4. Statistics of NSDL subset used for classification task with five categories.

This first classification task is: given the *title*, *content* and *description* of a record, predict the *audience* field value of the record from one the five given categories. In experiments, we use 5-fold cross validation and compare SRM with kNN (k-nearest neighbors with $k = 5$) and LibSVM (Chang & Lin, 2001), a publicly available implementation of support vector machine. (By building $n(n-1)/2$ SVMs, LibSVM can do multiple class classification having n categories quite well.) Table 5 shows thee three approaches' average recall rates for each category and overall average precisions for this classification task on the *audience* field from 5-fold runs. Higher value means better. It can be observed that SRM outperformed kNN (especially for the *middle school* category where SRM achieved an average recall rate of 0.843 while kNN achieves 0.641), but not decisively. The SVM learning approach solidly beat both of the other two, by a large degree and with statistical significance.

		5-NN		SVM		SRM	
		avg	stdev	avg	stdev	avg	stdev
recall	High school	0.874	0.010	0.988	0.004	0.862	0.006
	Middle school	0.641	0.034	0.965	0.008	0.843	0.026
	Undergraduate	0.518	0.153	0.870	0.087	0.606	0.184
	Graduate or Research	0.580	0.197	0.649	0.180	0.449	0.078
	Elementary school	0.524	0.061	0.873	0.030	0.588	0.055
overall precision		0.792	0.010	0.970	0.003	0.831	0.011

Table 5. Average Recalls and Precisions of SRM, 5-NN and SVM

We highlight these results to demonstrate that in some cases SRM will not be the appropriate solution. It turns out that by the way that evaluation set was constructed, all five possible values had a large number of non-overlapped instances in the corpus: *graduate or research* had the fewest at 69 and *high school* had the most at 5,210. Furthermore we deleted any instance in the original NSDL collection which belongs to more than one of these five categories, i.e. this learning task has been simplified to a single-labeled classification problem. Therefore a state-of-the-art learning approach such as SVMs should do well in this single-labeled learning scenario with so much training data, and doing well on a few large classes will obscure poorer results on smaller classes.

To confirm this hypothesis, we returned to the multi-labeled learning scenario on the *subject* field with 211 different values which appear most frequently in another subset of NSDL

collection. This subset includes all the records that have all five of the *title*, *content*, *description*, *subject* and *audience* fields – overall there are 11,596 this type of records. We followed the ranking approaches for multi-labeled learning and ranked (or selected) the *subject* values according to the posteriors of a given instance belonging to each *subject* value. We still utilized LibSVM which contains a probabilistic SVM for this ranking task. Then we did 5-fold cross validation and calculated the per-value average error rates for the SVM and SRM approaches, as a function of the number of instances of the value there were. Errors are measured as the proportion of *incorrect* labels that are ranked higher than the one being measured. Figure 1 and Table 6 show, for example, that if a value occurs 20-30 times, the SVM error rate is 31% compared to only 22% for the SRM approach; on the other hand, with 160-200 instances, the error rates are 5% and 12%, respectively. A similar experiment with 42 audience values yielded the same trend though with fewer values the error crossover happened at 30.

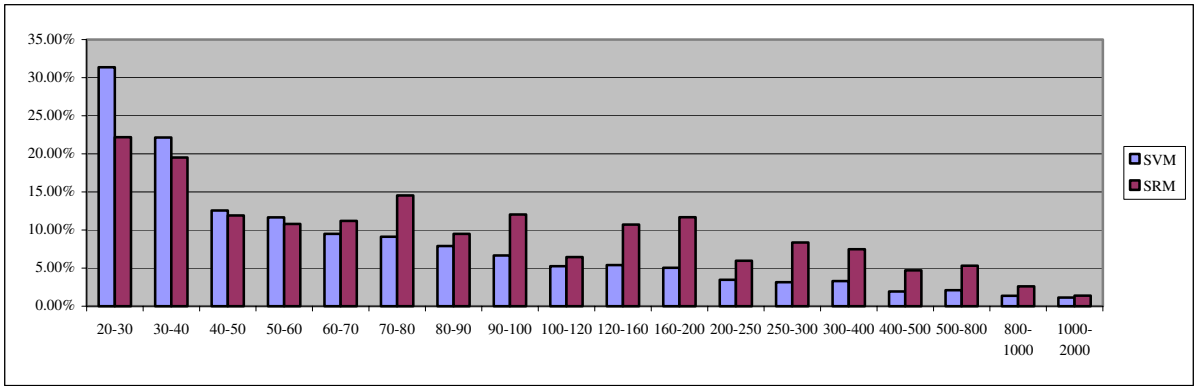

Figure 1. Average *error* rates for the SRM and SVM approaches to selecting the *subject* field values, as a function of the number of instances of a subject label there are in the corpus.

Num of instances		20-30	30-40	40-50	50-60	60-70	70-80
SVM	avg	31.37%	22.15%	12.57%	11.66%	9.49%	9.13%
	stdev	2.58%	2.23%	2.24%	2.16%	2.37%	3.33%
SRM	avg	22.19%	19.51%	11.91%	10.80%	11.19%	14.55%
	stdev	2.25%	2.71%	1.74%	1.78%	3.51%	1.67%
Num of instances		80-90	90-100	100-120	120-160	160-200	200-250
SVM	avg	7.90%	6.65%	5.25%	5.40%	5.04%	3.46%
	stdev	4.75%	3.22%	0.87%	2.52%	1.54%	1.46%
SRM	avg	9.50%	12.04%	6.46%	10.72%	11.68%	5.96%
	stdev	1.06%	1.79%	0.86%	1.70%	1.35%	1.37%
Num of instances		250-300	300-400	400-500	500-800	800-1000	1000-2000
SVM	avg	3.15%	3.29%	1.94%	2.09%	1.37%	1.12%
	stdev	1.03%	0.93%	0.96%	0.86%	1.12%	0.38%
SRM	avg	8.37%	7.48%	4.70%	5.32%	2.60%	1.39%
	stdev	1.05%	1.50%	0.24%	0.21%	0.47%	0.06%

Table 6. Averages and standard deviations of the *error* rates for the SRM and SVM approaches to selecting the *subject* field values.

Furthermore, we present the statistics for the number of instances of different *subject* values in original NSDL collection in Table 7. It can be observed that more than 90% of *subject* values have less than 60 instances. This indicates that for discovering missing values in the NSDL

collection for a free-text field like *subject*, where there are a great number of different values (37,385 in *subject* field) and a large portion of values only appear in a small number of instances, SRM is a much better choice than SVM.

Bin	0-10	10-20	20-30	30-40	40-50	50-60	60-70
Frequency	28724	2476	1110	658	476	355	264
Cumulative%	76.83%	83.46%	86.43%	88.19%	**89.46%**	**90.41%**	91.11%
Bin	70-80	80-90	90-100	100-120	120-160	160-200	200-250
Frequency	210	176	187	268	348	270	203
Cumulative%	91.68%	92.15%	92.65%	93.36%	94.29%	95.02%	95.56%
Bin	250-300	300-400	400-500	500-800	800-1000	1000-2000	>2000
Frequency	158	249	172	309	114	248	410
Cumulative%	95.98%	96.65%	97.11%	97.94%	98.24%	98.90%	100.00%

Table 7. Statistics for the number of instances of different *subject* values.

As a final test, we moved to an environment where the SVM training approach was prohibitively expensive in terms of time. For each test record, we ask the system to select a subset of *subject* field values from the 37,385 possible values in the corpus. That number of possible values is sufficiently large that we did not even consider implementing and training an SVM for each one; we just report results for the SRM approach. We also ask the system to select one or more of the 119 values of the *audience* field.

To evaluate this, we randomly selected 1,122 records (10% of our earlier set) that had all five fields. For each record, we used the *content*, *description*, and *title* fields to predict values of the *audience* or *subject* fields (separately). We used the remaining 655,870 records as training data, though it is important to note: 96.5% of the records were missing the *audience* field entirely and 23% lacked a *subject* field; furthermore some features were missing in the training data, e.g. 22% of the records were missing *description* field and 86% lacked a *content* field. Each test record could have multiple values of the field (on average, records contain 12 *subject* values and four *audience* values). The system's output was a ranked list of values for each field. Table 8 presents some concrete run results of predicting missing values by SRM in this experiment.

We evaluated SRM's performance using standard information retrieval measures based on where in the ranked list the *correct* values occur. Table 9 presents the evaluation results. An interesting measure is "P@5" showing that almost 2/3 of the audience values and just under half of the subject values listed in the top five items suggested were correct. The "R-precision" value measures the proportion of correct suggestions in the top *R* listed, where *R* is the actual number of suggestions that would have ideally been found. For example, if a record has 8 subjects assigned, then on average the top 8 suggestions would have included 5-6 (70%) that were correct. These are incredibly good results for a highly errorful and incomplete database of field values with very little training data.

Examples		Term Frequency	True Terms	$R_{subject}(v)$	Ordered term lists
Subject	1	3	wave	0.145	wave
		2	interference	0.132	physics
		1	physics	0.053	probable
		1	quantum	0.051	interference
		1	tutorial	0.039	quantum
		1	particle	0.037	tutorial
		1	slit	0.028	travel
		1	probable	0.022	slit
	2	1	astronomy	0.348	astronomy
		1	historic	0.175	general
		1	myth	0.174	historic
		1	legend	0.173	constellate
		1	constellate	0.006	physics
	3	1	science	0.182	calculus
		1	theory	0.137	variable
		1	precalculus	0.137	single
		1	calculus	0.0167	science
		1	linear	0.0164	multivariable
		1	algebra	0.0162	geometry
		1	number	0.0131	compute
Audience	1	1	teach	0.518	learn
		1	learn	0.283	teach
				0.002	school
	2	1	teach	0.238	teach
		1	researcher	0.235	learn
				0.104	researcher
	3	1	undergraduate	0.265	level
		1	level	0.259	undergraduate
		1	professional	0.231	professional

Table 8. Some examples of employing SRM for discovering missing *subject* and *audience* field values. For each given record, Column 3 shows the *true* field values of that record, Column 5 and Column 4 shows top-n term lists returned by SRM (cut by the number of *true* field values for subject field and by 3 for audience field) and their corresponding probabilities in relevance models, respectively.

	Audience	Subject
AMAP	0.8636	0.7451
P@5	0.6422	0.4617
P@20	0.1847	0.1496
R-precision	0.8259	0.7010

Table 9. Evaluation of SRM for discovering missing values in the NSDL collection.

6. Conclusions

In this paper, we developed the Structured Relevance Model (SRM) for the problem of discovering multiple missing field values in the NSDL collection. This 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. SRM is built on one hypothetical generative model for generating semi-structured

records, which follows the *kernel-density* approach and leverages training samples directly for representing probabilistic density function.

We validated the SRM approach for inference firstly on a small scale, binary movie label prediction task with IMDb records, and then on a multiple missing values selection task with a large archive of the NSDL repository. Experiments on IMDb records show that SRM matched three state-of-the-art relational learning approaches on the movie label prediction tasks by achieving an average accuracy of 79.5%. Experiments on the NSDL collection show that compared with SVM, SRM is highly effective at discovering possible values for free-text fields even with quite modest amounts of training data. In one experiment with all NSDL records, SRM performed very well for the missing values discovery task where there are a huge number of instances and missing fields: SRM brought 5-6 correct missing subject values into the top 8 and achieved an average precision of 74.5% for selecting the subject field values.

The next step of our research is to adopt SRM for answering complex structured queries over a semi-structured collection with corrupt and missing field values by leveraging the discovered relevant missing values. For example, by seeing 'elementary' in the *audience* field of a user's input query, SRM may be able to retrieve records having '*K-4*', '*second grade*', '*learner*' or other related values in the audience field, which the user is *plausibly* interested in. Furthermore we will empirically test SRM's performance on more IR tasks.

Our results on the Internet movie database were much better than we anticipated and we are investigating that more carefully. We are interested in extending that work to require less supervision during the "flattening" process (when the relational structure is stripped). Can we predict attribute values by "flattening" to a certain distance on the graph without regard to the structure?

In addition we have also begun explorations toward using inferred values to help a user browse a collection when starting from some structured information---e.g., given values for two fields, what values are probable for other fields.

It is worth highlighting that SRM is based on relevance modeling (Lavrenko & Croft, 2003), a technique that has proven highly effective in a number of diverse applications in IR field. SRM suggests one promising way to introduce this IR technique for machine learning task, which can not only be used for typical multi-label classification problems, but also for the very difficult multi-label learning task where there are a huge number of different categories.

Acknowledgements

We thank David Jensen for his valuable comments and help on this work. 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

Chang, C.C. & Lin, C.J. (2001). Libsvm: a library for support vector machines. http://www.csie.ntu.edu.tw/ cjlin/libsvm.

Denoyer, L. & Gallinari, P. (2004). Bayesian Network Model For Semi-Structured Document Classification. In *Information Processing and Management*. Volume 40, Issue 5 (September), 807—827, 2004.

Diligenti, M. & Gori, M. & Maggini, M. & Scarselli, F. (2001). Classification of HTML documents by Hidden Tree-Markov Models. In *Proceedings of ICDAR 2001 (pp. 849-853)*. Seattle, WA, USA: IEEE Computer Society

Elisseeff, A. & Weston, J. (2002). A kernel method for multi-labelled classification. In *Advances in Neural Information Processing Systems 14* (pp. 681--687). Cambridge, MA: MIT Press.

Friedman, N. & Getoor, L. & Koller, D. & Pfeffer, A. (1999). Learning Probabilistic Relational Models. In *IJCAI*, (pp. 1300--1390). Stockholm, Sweden.

Godbole, S. & Sarawagi, S. (2004). Discriminative methods for multi-labeled classification. In *Proceedings of PAKDD 2004 (pp.22--30)*. Sydney, Australia: Springer

Grabs, T. & Schek, H. (2002). ETH Zürich at INEX: Flexible Information Retrieval from XML with PowerDB-XML. In *Proceedings of the Initiative for the Evaluation of XML retrieval (INEX) 2002 Workshop* (pp. 141--148). Schloss Dagstuhl:ERCIM.

Jensen, D. & Neville, J. (2002). Linkage and autocorrelation cause bias in relational feature selection. In *Proceedings of the Nineteenth ICML*. Sydney, Australia: Morgan Kaufmann.

Lafferty, J. & McCallum, A. & Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML2001* (pp. 282-289). San Francisco, CA, USA: Morgan Kaufmann.

Lavrenko, V. & Croft, W.B. (2001). Relevance-based language models. In *Proceedings of ACM SIGIR* (pp. 120--127). New Orleans, Louisiana, USA.

Lavrenko, V. & Croft, W. B. (2003). Relevance Models in Information Retrieval. In *Language Modeling for Information Retrieval* (pp. 11--56). Dordrecht, The Netherlands: Kluwer Academic Publishers.

McCallum, A. (1999). Multi-label text classification with a mixture model trained by EM. In *Proceedings of AAAI'99 Workshop on Text Learning*. Orlando, Florida, USA.

Neville, J. & Jensen, D. (2003). Collective classification with relational dependency networks. In *Proceedings of the 2nd Multi-Relational Data Mining Workshop, 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Washington, DC, USA.

Neville, J. & Jensen, D. & Friedland, L. & Hay, M. (2003). Learning relational probability trees. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Washington, DC, USA.

Neville, J. & Jensen, D. & Gallagher, B. (2003). Simple estimators for relational Bayesian classifiers. In *Proceedings of The Third IEEE International Conference on Data Mining* (pp. 608--612). Melbourne, Florida, USA: IEEE Computer Society Press.

Ponte, J.M. & Croft, W.B. (1998). A language modeling approach to information retrieval. In *Proceedings of ACM SIGIR* (pp. 275--281). Melbourne, Australia.

Rousu, J. & Saunders, C. & Szedmak, S. & Shawe-Taylor, J. (2006). Kernel-Based Learning of Hierarchical Multilabel Classification Models. *Journal of Machine Learning Research*, 7(Jul),1601--1626, 2006.

Schapire, R.E. & Singer, Y. (2000). Boostexter: A boosting-based system for text classification. *Machine Learning*, 39, 135--168.

Silverman, B. (1986). *Density Estimation for Statistics and Data Analysis*. pages 75--94. CRC Press.

Taskar, B. & Abbeel, P. & Koller, D. (2002). Discriminative probabilistic models for relational data. In *Proceedings of the 18th Annual Conference on Uncertainty in Artificial Intelligence (UAI-02)* (pp. 485--49). San Francisco, CA, USA: Morgan Kaufmann.

Taskar, B. & Segal, E. & Koller, D. (2001). Probabilistic classification and clustering in relational data. In *IJCAI*, (pp. 870--876). Seattle, Washington, USA.

Ueda, N. & Saito, K. (2003). Parametric mixture models for multi-labeled text. In *advances in Neural Information Processing Systems 15*. Cambridge, MA: MIT Press.

Zhu, S. & Ji, X. & Xu, W. & Gong, Y. (2005). Multi-labelled Classification Using Maximum Entropy Method. In *Proceedings of the Twenty-Eighth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 274--281). Salvador, Brazil.