Learning Extractors from Unlabeled Text using Relevant Databases

Kedar Bellare and **Andrew McCallum**
Department of Computer Science
140 Governor's Drive
University of Massachusetts
Amherst, MA 01003-4601
{kedarb,mccallum}@cs.umass.edu

Abstract

Supervised machine learning algorithms for information extraction generally require large amounts of training data. In many cases where labeling training data is burdensome, there may, however, already exist an incomplete database relevant to the task at hand. Records from this database can be used to label text strings that express the same information. For tasks where text strings do not follow the same format or layout, and additionally may contain extra information, labeling the strings completely may be problematic. This paper presents a method for training extractors which fill in missing labels of a text sequence that is partially labeled using simple high-precision heuristics. Furthermore, we improve the algorithm by utilizing labeled fields from the database. In experiments with BibTeX records and research paper citation strings, we show a significant improvement in extraction accuracy over a baseline that only relies on the database for training data.

Introduction

The Web is a large repository of data sources where majority of the information is represented as unstructured text. Efficiently mining and querying such sources requires large-scale integration of unstructured resources into a structured database. A first step towards achieving this goal involves extraction of record-like information from unstructured and unlabeled text. Information extraction (IE) approaches address the problem of segmenting and labeling unstructured text to populate a database with a pre-defined schema.

In this paper, we mainly deal with information extraction from *textual records*. Textual records are contiguous fragments of text present in unstructured text documents that resemble fields of a database record. In the context of restaurant addresses, an instance of a textual record is: *"katsu. 1972 hillhurst ave. los feliz, california, 213-665-1891"* which can be segmented as [name= *"katsu."*, address= *"1972 hillhurst ave."*, city= *"los feliz"*, state= *"california"*, phone= *"213-665-1891"*]. Examples of textual records include citation strings found in research papers and contact addresses found on person homepages. Pattern matching and rule-based approaches for IE that only depend on delimiters and other structure and font-based cues for segmenting such data are prone to failure as these markers are generally not reliable. In addition, the varying order of fields rendered in text further exacerbates extraction. Algorithms based on supervised machine learning such as conditional random fields (CRFs) (Lafferty, McCallum, & Pereira 2001; Peng & McCallum 2004; McCallum 2003; Sarawagi & Cohen 2005) and hidden Markov models (HMMs) (Rabiner 1989; Seymore, McCallum, & Rosenfeld 1999; Freitag & McCallum 1999) are popular approaches for solving such tasks. These approaches, however, require labeled training data, such as annotated text, which is often scarce and expensive to produce. In many cases, there already exists a database with data related to the expected input of the algorithm, and structure relevant to the desired output. This database may be incomplete and noisy, and be missing the surrounding contexts that act as reliable indicators for recognizing these data as rendered in unlabeled data, however, the database can nonetheless act as a significant source of supervised guidance.

Previous work using databases to train information extractors has taken one of two simpler approaches. The first attempts to train directly from the database alone – missing information about typical transformations and context that occur in the rendering of that data. For example, the method proposed by Agichtein & Ganti (2004) trains a separate hidden Markov model for each field directly from database field string values. The second approach uses hand-built heuristic rules to apply database field labels to unlabeled data, on which training is then performed. The work of Geng & Yang (2004) is one example. While correctly capturing the context of data fields as they are rendered in the wild, this approach discards sequences that are not fully labeled. When noisy and complex transformations are involved in the conversion of a record to text such an approach may not work well.

We build upon the framework of conditional random fields to address these challenges. We present a method that only requires *some* of the tokens in the text to be labeled with high-precision. The input to the algorithm is a matching[1] database record and textual record pair. An ex-

[1]Associations among database and text can typically be easily

ample of a matching database record for the text *"katsu. 1972 hillhurst ave. los feliz, california, 213-665-1891"* can be [name=*"restaurant katsu"*, address=*"n. hillhurst avenue"*, city=*"los angeles"*, state=*""*, phone=*"213-665-1891"*]. Given such a pair, a high-precision partial labeling of the text sequence can be constructed easily. A simple linear-chain CRF is trained with an expected gradient procedure to fill in missing labels of the text sequence. Similar to Agichtein & Ganti (2004), we also explore the use of labeled field information in the database to further improve our model. Employing a CRF during extraction enables us to utilize multiple overlapping and rich features of the input text in comparison with HMM-based techniques. Once an extractor is trained, it is used to segment and label unseen text sequences which are then added as records to the original database.

We present experimental results on a citation extraction task using real-world BibTeX databases. Our results demonstrate a significant improvement over a baseline that only relies on database records during training. For the baseline we implemented an enhanced version of the state-of-the-art system presented in (Agichtein & Ganti 2004) . We generate "pseudo"-textual records from a database by artificially concatenating fields in the order in which they may appear in the real-world. Furthermore, real-world noise (e.g. spelling errors, word and field deletions) is artificially added to the database record before concatenation. Given such a sequence that is fully-labeled using database field names, a linear-chain CRF is trained on the data. In our experiments, we show a 21% improvement in accuracy over this baseline of an extractor trained only on partially-labeled text sequences. Furthermore, adding labeled supervision in the form of database fields produces an additional 6% absolute improvement in performance.

Related Work

Databases have been used in information extraction (IE) systems either for labeling text in the real world (Geng & Yang 2004; Craven & Kumlien 1999) or as sources of weakly-labeled data (Agichtein & Ganti 2004; Seymore, McCallum, & Rosenfeld 1999).

Gene and protein databases are widely used for IE from biomedical text. An automated process is employed in (Craven & Kumlien 1999) to label biomedical abstracts with the help of an existing protein database. The labeling procedure looks for exact mentions of database tuples in text and does not deal with missing or noisy fields. Geng & Yang (2004) suggest the use of database fields to heuristically label text in the real world and then train an extractor only on data labeled completely with high-confidence. Producing a heuristic labeling of the entire text with high confidence may not be possible in general. We address this limitation in our work by requiring only a subset of tokens to be labeled with high-precision. Other approaches (Sarawagi & Cohen 2005; Sarawagi & Cohen 2004) employ database or dictionary lookups in combination with similarity measures to add features to the text sequence. Although these features are helpful at training time, learning algorithms still require labeled data to make use of them.

Recent work on reference extraction with HMMs (Seymore, McCallum, & Rosenfeld 1999) has used BibTeX records along with labeled citations to derive emission distributions for the HMM states. However, learning transition probabilities for the HMM requires labeled examples to capture the structure present in real data. Some IE methods trained directly on database records (Agichtein & Ganti 2004), encode relaxations in the finite-state machine (FSM) to capture the errors that may exist in real world text. Only a few common relaxations such as token deletions, swaps and insertions can be encoded into the model effectively. Because our model is trained directly on the text itself, we are able to capture the variability present in real-world text.

Extraction framework

In this section, we present an overview of the framework we employ to learn extractors from pairs of corresponding database-text records.

A linear-chain conditional random field (CRF) constitutes the basic building block in all our algorithms. The linear-chain CRF is an undirected graphical model consisting of a sequence of output random variables $\mathbf{y}$ connected to form a linear-chain under first-order Markov assumptions. CRFs are trained to maximize the conditional likelihood of the output variables $\mathbf{y} : \langle y_1 y_2 \dots y_T \rangle$ given the input random variables $\mathbf{x} : \langle x_1 x_2 \dots x_T \rangle$. The probability distribution $p(\mathbf{y}|\mathbf{x})$ has an exponential form with feature functions f encoding sufficient statistics of $(\mathbf{x}, \mathbf{y})$. The output variables generally take on discrete states from an underlying finite state machine where each state is associated with a unique label $y \in \mathcal{Y}$. Hence, the conditional probability distribution $p(\mathbf{y}|\mathbf{x})$ is given by,

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left(\sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(y_{t-1}, y_t, \mathbf{x}, t) \right) \quad (1)$$

where T is the length of the sequence, K is the number of feature functions f_k and λ_k are the parameters of the distribution. $Z(\mathbf{x})$ is the partition function given by, $Z(\mathbf{x}) = \sum_{\mathbf{y}' \in \mathcal{Y}^T} \exp\left(\sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(y'_{t-1}, y'_t, \mathbf{x}, t) \right)$.

When there are missing labels $\mathbf{h} = \langle h_1 h_2 \dots h_m \rangle$, their effect needs to be marginalized out when calculating the conditional probability distribution $p(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{h} \in \mathcal{Y}^m} p(\mathbf{h}, \mathbf{y}|\mathbf{x})$.

The feature functions in a CRF model allow additional flexibility to effectively take advantage of complex overlapping features of the input. These features can encode binary or real-valued attributes of the input sequence with arbitrary lookahead.

The procedure for inference in linear-chain CRFs is similar to those used in HMMs. Baum-Welch and Viterbi algorithms are easily extended as described in Lafferty, McCallum, & Pereira (2001). During training, the parameters of the CRF are set to maximize

created through heuristic methods or the use of an inverted index. These associations may also be created by asking the user for a match/non-match label.

the conditional likelihood of the training set $\mathcal{D} = \{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), (\mathbf{x}^{(2)}, \mathbf{y}^{(2)}), \ldots, (\mathbf{x}^{(N)}, \mathbf{y}^{(N)})\}$.

Data format

In this section, we briefly work through an example of a database containing restaurant records and a relevant textual record found on a restaurant review website.

The input to our system is a database consisting of a set of possibly noisy database (DB) records and corresponding textual records that are realizations of these DB records. For example, a matching record-text pair can be [name=*"restaurant katsu"*, address=*"n. hillhurst avenue"*, city=*"los angeles"*, state=*""*, phone=*"213-665-1891"*] and the text rendering *"katsu. 1972 hillhurst ave. los feliz, california, 213-665-1891"*. Notice that the field state is missing in the DB record. We assume a tokenization of both the record and the text string based on whitespace characters. Let $\mathbf{x} = \langle x_1 x_2 \ldots x_T \rangle$ represent the sequence of tokens in the text string under such a tokenization. We are not given a label sequence corresponding to the input sequence $\mathbf{x}$.

Let $\mathbf{x}'$ be the DB record and $\mathbf{x}'[i]$ indexes the i^{th} field in the record. Let $y'[i]$ be the label corresponding to the token sequence $\mathbf{x}'[i]$ where all the labels in the sequence have the same value. If some of the fields in the DB record are empty then the corresponding record and label sequence are also empty.

We build robust statistical models to learn extractors by leveraging label information from DB records. Using the named fields of a DB tuple we unambiguously label some of the tokens in the matching text string with high confidence. To construct a partial labeling of the text we use heuristics that include exact token match or strong similarity match based on character edit distance (e.g. Levenshtein or character n-grams). In addition, we require that there be a unique match of a record token within the text sequence. The final labels produced have high precision although many of the tokens may remain unlabeled, leading to low recall.

A partial labeling of the text is denoted by $\mathbf{y} = \langle y_{t_1} y_{t_2} \ldots y_{t_m} \rangle$ where $t_1, t_2, \ldots t_m$ are the positions in the sequence where labels are observed. In the gaps of the observed sequence $\mathbf{y}$, there exists a hidden label sequence $\mathbf{h} = \langle h_{t'_1} h_{t'_2} \ldots h_{t'_n} \rangle$ with unobserved label positions $t'_1, t'_2, \ldots t'_m$. In the above example, *"katsu", "hillhurst", "los"* and *"213-665-1891"* would be labeled by the high-precision heuristics with name, address, city and phone respectively. The tokens *"1972", "ave."* and *"feliz"* would remain unlabeled. If there were an extra token *"los"* in the address field (e.g. *"los n. hillhurst avenue"*) or in the text string (e.g. *"los katsu 1972 hillhurst ..."*), then the tokens *"los"* in the various fields would also remain unlabeled.

CRF Training with Missing Labels

For a linear chain CRF with an underlying FSM we assume that each state has a corresponding unique output label $y \in \mathcal{Y}$. In the case of restaurant addresses, the states can be name, address, city, state and phone with a fully connected state machine. Given an input sequence $\mathbf{x}$ with a partially observed label sequence $\mathbf{y} = \langle y_{t_1} y_{t_2} \ldots y_{t_m} \rangle$ and a hidden label sequence $\mathbf{h} = \langle h_{t'_1} h_{t'_2} \ldots h_{t'_n} \rangle$, the conditional probability $p(\mathbf{y}|\mathbf{x})$ is given by,

$$\begin{aligned} p(\mathbf{y}|\mathbf{x}) &= \sum_{\mathbf{h}} p(\mathbf{y}, \mathbf{h}|\mathbf{x}) \qquad (2) \\ &= \frac{1}{Z(\mathbf{x})} \sum_{\mathbf{h}} \exp(\sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(z_{t-1}, z_t, \mathbf{x}, t)) \end{aligned}$$

where

$$z_t = \begin{cases} y_{t_i} & \text{for} \quad t = t_i \\ h_{t'_j} & \text{for} \quad t = t'_j \end{cases}$$

We could use EM to maximize conditional data likelihood (Equation 2). However, we employ a direct gradient ascent procedure similar to expected conjugate gradient suggested by Salakhutdinov, Roweis, & Ghahramani (2003),

$$\begin{aligned} \log p(\mathbf{y}|\mathbf{x}) &= \log(\sum_{\mathbf{h}} \exp(\sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(z_{t-1}, z_t, \mathbf{x}, t))) \\ &\quad - \log(\sum_{\mathbf{y}'} \exp(\sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(y'_{t-1}, y'_t, \mathbf{x}, t))) \\ \frac{\partial \log p(\mathbf{y}|\mathbf{x})}{\partial \lambda_k} &= \sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{y}, \mathbf{x}) \sum_{t=1}^{T} f_k(z_{t-1}, z_t, \mathbf{x}, t) \\ &\quad - \sum_{\mathbf{y}'} p(\mathbf{y}'|\mathbf{x}) \sum_{t=1}^{T} f_k(y'_{t-1}, y'_t, \mathbf{x}, t) \end{aligned}$$

For the dataset of text strings that are partially labeled using heuristics $\mathcal{D} = \{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), (\mathbf{x}^{(2)}, \mathbf{y}^{(2)}), \ldots (\mathbf{x}^{(N)}, \mathbf{y}^{(N)})\}$, we maximize the conditional log-likelihood.

$$\mathcal{L}(\Lambda = \langle \lambda_1 \ldots \lambda_K \rangle; \mathcal{D}) = \sum_{i=1}^{N} \log\left(p(\mathbf{y}^{(i)}|\mathbf{x}^{(i)})\right) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2}$$

where σ is the gaussian variance used for regularization. Thus,

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \lambda_k} &= \sum_{i=1}^{N} \sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{y}^{(i)}, \mathbf{x}^{(i)}) \sum_{t=1}^{T} f_k(z_{t-1}, z_t, \mathbf{x}^{(i)}, t) \\ &\quad - \sum_{i=1}^{N} \sum_{\mathbf{y}'} p(\mathbf{y}'|\mathbf{x}^{(i)}) \sum_{t=1}^{T} f_k(y'_{t-1}, y'_t, \mathbf{x}^{(i)}, t) \\ &\quad - \frac{\lambda_k}{\sigma^2} \qquad (3) \end{aligned}$$

The expected feature counts in Equation (3) are easily calculated using the forward-backward procedure (Culotta & McCallum 2004; McCallum 2003). Thus, given a partially labeled data set of textual records we can maximize the parameters such that the conditional likelihood of the observed labels is maximized. In the process, missing labels are filled in such that they best explain the observed data and hence a partially labeled text record is fully labeled.

To maximize log-likelihood we use a quasi-Newton L-BFGS optimization. Since there are missing labels, the optimization is no longer convex and sub-optimal local minima are possible. Some of the parameters may be initialized to reasonable values to attain a good local optimum, but we did not encounter problems of local maxima in practice. The parameters learned by the model are used at inference time to decode new text sequences.

Henceforth, we call this model as **M-CRF** for missing label linear-chain CRF.

CRF Training with the DB

The database provided as input to the system generally contains extra information that is not present in the partially labeled text sequences such as:

- Additional fields that are not rendered in text.
- Training data for modeling transitions within the same field.

Therefore, we train a CRF classifier on individual fields of a DB record $\mathbf{x}' = \{\mathbf{x}'[1], \mathbf{x}'[2], \ldots, \mathbf{x}'[n]\}$. The state machine we employ is the same as the FSM used in the **M-CRF** model and is trained to distinguish the field that a particular string belongs to. The labels at training time correspond to column names in the DB schema $\mathbf{y}' = \{y'[1], y'[2], \ldots, y'[n]\}$.

For a particular field $\mathbf{x}'[i] = \langle x'_1 x'_2 \ldots x'_T \rangle$ we replicate the label $y'[i]$ for T timesteps and construct a new label sequence $\mathbf{y}'[i] = \langle y'_1 y'_2 \ldots y'_T \rangle$. A linear-chain CRF is then used as a classifier where the conditional probability for an individual field $p(\mathbf{y}'[i] \mid \mathbf{x}'[i])$ is given by,

$$p(\mathbf{y}'[i] \mid \mathbf{x}'[i]) = \frac{1}{Z(\mathbf{x}'[i])} \exp(\sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(y'_{t-1}, y'_t, \mathbf{x}'[i], t))$$

and the conditional probability of a complete DB record $\mathbf{x}'$ is given by,

$$p(\mathbf{y}'|\mathbf{x}') = \prod_{i=1}^{n} p(\mathbf{y}'[i] \mid \mathbf{x}'[i])$$

Note that we do not model connections between different fields and treat each field independently. Given a database of records $\mathcal{D}' = \{\mathbf{x}'^{(1)}, \mathbf{x}'^{(2)}, \ldots, \mathbf{x}'^{(M)}\}$ and a fixed schema $\mathbf{y}' = \{y'[1], y'[2], \ldots, y'[n]\}$, we optimize the parameters Λ to maximize the conditional likelihood of the data $\mathcal{D}'$,

$$\begin{aligned}\mathcal{L}(\Lambda = \langle \lambda_1 \ldots \lambda_K \rangle; \mathcal{D}') &= \sum_{i=1}^{M} \log \left(p(\mathbf{y}'|\mathbf{x}'^{(i)}) \right) \\ &= \sum_{i=1}^{M} \sum_{j=1}^{n} \log \left(p(\mathbf{y}'[j]|\mathbf{x}'^{(i)}[j]) \right)\end{aligned}$$

As there are no hidden variables or missing labels this learning problem is convex and optimal parameters can be obtained though an L-BFGS procedure. During decoding, when a complete text record needs to be segmented we apply the Viterbi procedure to produce the best labeling of the text.

Since we do not learn parameters for transitions between fields, this approach, henceforth know as **DB-CRF**, may perform poorly when applied in isolation. However, the current approach can be used to initialize the parameters of the **M-CRF** model and then the **M-CRF** model can be further trained on partially labeled text sequences. We call such an approach **DB+M-CRF**.

Experiments

We present experimental results on a reference extraction task which uses a database of BibTeX records and research paper citations to learn an extractor. The Cora data set (McCallum *et al.* 2000) contains segmented and labeled references to 500 unique computer science research papers. This data set has been used in prior evaluations of information extraction algorithms (McCallum *et al.* 2000; Seymore, McCallum, & Rosenfeld 1999; Han *et al.* 2003; Peng & McCallum 2004). We collected a set of BibTeX records from the web by issuing queries to a search engine with keywords present in the citation text. We gathered matching BibTeX records for 350 citations in the Cora data set. We fixed the set of labels to the field names in references, namely, *author, title, booktitle, journal, editor, volume, pages, date, institution, location, publisher and note*. We throw away certain extra labels in the BibTeX such as *url* and *isbn*. Other field names like *month* and *year* are converted into *date*. Finally the data set formed consists of matching record-text pairs where the labels on the text are used only at test time. We perform 7-fold cross-validation on the data set with 300/50 training and test examples respectively.

We use a variety of features to enhance system performance. Regular expressions are used to detect whether a token contains all characters, all digits or both digits and characters (e.g. ALLCHAR, ALLDIGITS, ALPHADIGITS). We also encode the number of characters or digits in the token as a feature (e.g. NUMCHAR=3, NUMDIGITS=1). Other domain-specific patterns for dates, pages and URLs are also helpful (DOM). We also employ identity, suffixes, prefixes and character n-grams of the token (TOK). Another class of features that are helpful during extraction are lexicon features (LEX). Presence of a token in lexicon such as "Common Last Names", "Publisher names", "US Cities" as binary features help improve the accuracy of the model. Finally, binary features at certain offsets (e.g. +1, -2) and various conjunctions can be added to the model (OFFSET). Thus, a conditional model lets us use these arbitrary helpful features that could not be exploited in a generative model.

We compare our models against a baseline that only relies on the database fields for training data. Instead of the method suggested by Agichtein & Ganti (2004) in which each field of the database is modeled separately, we actually simulate the construction of a textual record from a given DB record. A CRF extractor is then trained using all these labeled records. The baseline model (**B-CRF**) uses labeled text data that is constructed by artificially concatenating database fields in an order that is observed in real-world text. This order is randomly chosen from a list of real-world

	B-CRF	M-CRF	DB-CRF	DB+M-CRF	GS-CRF
Overall acc.	64.1%	85.2%	47.4%	**91.9%**	96.7%
author	92.9	89.6	87.0	**97.5**	99.2
title	69.7	93.7	50.6	**96.9**	98.3
booktitle	65.1	88.3	51.8	**90.5**	96.0
journal	64.8	78.1	31.0	**83.8**	92.7
editor	11.7	1.4	0.0	**56.5**	90.4
volume	37.8	68.3	1.4	**83.7**	96.9
pages	65.8	74.9	22.5	**90.3**	98.0
date	62.6	75.5	1.2	**94.2**	98.2
institution	0.0	6.8	9.1	**20.4**	79.4
location	18.8	47.1	10.7	**80.0**	89.5
publisher	34.9	59.6	28.8	**84.2**	94.1
note	7.8	4.1	4.5	**8.2**	26.2
Average F-1	48.4	61.9	26.3	**78.7**	89.0

Table 1: Extraction results on the paired BibTeX-citation data set. Bold-faced numbers indicate the best model that is closest in performance to the gold-standard extractor.

orderings (provided by the user) that occur in text. Furthermore, delimiters are placed between fields while concatenating them. We further strengthen the baseline by modeling variations of the record as they appear in the text. We randomly remove delimiting information, delete tokens, insert tokens, introduce spelling errors and delete a field before concatenation to simulate noise in the real-world text. Each of these errors in injected with a probability of 0.1 during the concatenation phase. Given a set of fully-labeled noisy token sequences a linear-chain CRF is trained to maximize the conditional likelihood of the observed label sequences. The trained CRF is then used as an extractor at test time. Lastly, the gold-standard extractor (**GS-CRF**) is a CRF trained on labeled citations from the Cora data set.

The evaluation metrics used in the comparison of our algorithms against the baseline model are overall accuracy (number of tokens labeled correctly in all text sequences ignoring punctuation) and per-field F1. Table 1 shows the results of various algorithms applied to the data set. The **M-CRF** model clearly provides an improvement of 21% over the baseline **B-CRF** in terms of overall accuracy. Using **DB-CRF** in isolation without labeled text data does worse than the baseline. However, initializing the model with parameters learned from **DB-CRF** to train the **DB+M-CRF** model provides a further boost of 6% absolute improvement in overall accuracy over **M-CRF** model. All these differences are significant at $p = 0.05$ under 1-tailed paired t-test. The **DB+M-CRF** model provides the highest performance and is indeed very close to the gold-standard performance of **GS-CRF**.

The effect of token-specific prefix/suffix (TOK), domain-specific (DOM), lexicon (LEX) and offset-conjunction features (OFFSET) for the **DB+M-CRF** model is explored in Table 2. In particular, we find that the TOK features are most helpful during extraction.

Table 3 shows the percentage of DB records that contain a certain field in our data set. In general, fields that are poorly represented in DB records such as *editor, institution* and *note* have a corresponding low F1 performance during extraction. Additionally, since the content of the *note* field is varied the extraction accuracy is further reduced.

Feature type	% Overall acc.
ALL	91.94
–OFFSET	91.95
–LEX	90.89^*
–DOM	90.72
–TOK	14.46^*

Table 2: Effect of removing OFFSET, LEX, DOM, TOK features one-by-one on the extraction performance of **DB+M-CRF** model. Starred numbers indicate significant difference over the row above it ($p = 0.05$).

The largest improvement in performance is observed for the *editor* field. During training, only a few partially labeled text sequences contain tokens labeled as *editor*. Due to lack of supervision the **M-CRF** model is unable to disambiguate between the *author* and *editor* field and therefore incorrectly labels the *editor* fields in the test data. However, by additionally utilizing *editor* fields only present *only* in the database and not in text the **DB+M-CRF** model provides an absolute improvement of 45% in F1 over the best baseline. Hence, an advantage of the DB is that it gives further supervision when there are extra fields in a record that are not present in the matching text.

The database also improves segmentation accuracy by modeling individual fields and their boundary tokens. This is especially noticeable for fields such as *booktitle* and *title* where starting tokens may be stop words that are generally left unlabeled by high-precision heuristics. We find that errors in labeling tokens by the **M-CRF** model on the *author/title*, *title/booktitle* and *author/booktitle* boundaries are correctly handled by the **DB+M-CRF** model.

Furthermore, by modeling confusing fields such as *volume, pages* and *date* in context, the **M-CRF** and **DB+M-**

Field name	% occurence in DB records
author	99.7
title	99.4
booktitle	43.9
journal	39.9
editor	5.7
volume	43.9
pages	82.2
date	100.0
institution	2.3
location	49.6
publisher	50.9
note	4.2

Table 3: Percentage of records containing a certain field.

CRF models provide an improvement over other baselines. Similar improvements are observed for *booktitle/journal* and *publisher/location*. The latter confusion is caused by city names present in *publisher* fields (e.g., *"New York"* in the field [publisher=*"Springer-Verlag New York, Inc."*]).

Hence, a combination of partially labeled text data and fully labeled DB records are helpful for extraction.

Conclusions

This paper suggests new ways of leveraging databases and text sources to automatically learn extractors. We obtain significant error reductions over a baseline model that only relies on the database for supervision. We find that a combination of partially labeled text strings and labeled database record strings are required to attain state-of-the-art performance.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval and in part by The Central Intelligence Agency, the National Security Agency and National Science Foundation under NSF grant #IIS-0326249. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

References

Agichtein, E., and Ganti, V. 2004. Mining reference tables for automatic text segmentation. In *KDD*, 20.

Craven, M., and Kumlien, J. 1999. Constructing biological knowledge bases by extracting information from text sources. In *ISMB*, 77–86. AAAI Press.

Culotta, A., and McCallum, A. 2004. Confidence estimation for information extraction. In *Proceedings of Human Language Technology Conference and North American Chapter of the Association for Computational Linguistics (HLT-NAACL). 2004. Boston, MA.*, volume 8.

Freitag, D., and McCallum, A. 1999. Information extraction with HMM and shrinkage. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI)*.

Geng, J., and Yang, J. 2004. Autobib: Automatic extraction of bibliographic information on the web. In *IDEAS*, 193–204.

Han, H.; Giles, C. L.; Manavoglu, E.; Zha, H.; Zhang, Z.; and Fox, E. A. 2003. Automatic document metadata extraction using support vector machines. In *JCDL*, 37–48.

Lafferty, J.; McCallum, A.; and Pereira, F. C. N. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 282.

McCallum, A.; Nigam, K.; Rennie, J.; and Seymore, K. 2000. Automating the construction of internet portals with machine learning. *Information Retrieval Journal* 3:127–163.

McCallum, A. 2003. Efficiently inducing features of conditional random fields. In *UAI*, 403–41. San Francisco, CA: Morgan Kaufmann.

Peng, F., and McCallum, A. 2004. Accurate information extraction from research papers using conditional random fields. In *HLT-NAACL*, 329–336.

Rabiner, L. R. 1989. A tutorial on hidden markov models and selected applications in speech processing. *IEEE* 17:257–286.

Salakhutdinov, R.; Roweis, S. T.; and Ghahramani, Z. 2003. Optimization with em and expectation-conjugate-gradient. In *ICML*, 672.

Sarawagi, S., and Cohen, W. W. 2004. Exploiting dictionaries in named entity extraction: combining semi-markov extraction processes and data integration methods. In *KDD*, 89.

Sarawagi, S., and Cohen, W. W. 2005. Semi-markov conditional random fields for information extraction. In *NIPS*. Cambridge, MA: MIT Press. 1185–1192.

Seymore, K.; McCallum, A.; and Rosenfeld, R. 1999. Learning hidden markov model structure for information extraction. In *Proceedings of the AAAI'99 Workshop on Machine Learning for Information Extraction*.