Multilingualism and Electronic Language Management: Proceedings of the 4th International MIDP Colloquium, September 2003, Bloemfontein, South Africa, Daelemans, W., du Plessis, T., Snyman, C. and Teck, L. (Eds.) pp.141-160, Van Schaik Pub., South Africa, 2005

Text Mining with Information Extraction

Raymond J. Mooney and Un Yong Nahm
Department of Computer Sciences,
University of Texas, Austin, TX 78712-1188
{mooney,pebronia}@cs.utexas.edu

Abstract

Text mining concerns looking for patterns in unstructured text. The related task of *Information Extraction* (IE) is about locating specific items in natural-language documents. This paper presents a framework for text mining, called DISCOTEX (Discovery from Text EXtraction), using a learned information extraction system to transform text into more structured data which is then mined for interesting relationships. The initial version of DISCOTEX integrates an IE module acquired by an IE learning system, and a standard rule induction module. In addition, rules mined from a database extracted from a corpus of texts are used to predict additional information to extract from future documents, thereby improving the recall of the underlying extraction system. Encouraging results are presented on applying these techniques to a corpus of computer job announcement postings from an Internet newsgroup.

1 Introduction

The problem of *text mining*, i.e. discovering useful knowledge from unstructured or semi-structured text, is attracting increasing attention [4, 18, 19, 21, 22, 27]. This paper suggests a new framework for text mining based on the integration of *Information Extraction* (IE) and Knowledge Discovery from Databases (KDD), a.k.a. *data mining*. KDD and IE are both topics of significant recent interest. KDD considers the application of statistical and machine-learning methods to discover novel relationships in large relational databases. IE concerns locating specific pieces of data in natural-language documents, thereby extracting structured information from free text. However, there has been little if any research exploring the interaction between these two important areas. In this paper, we explore the mutual benefit that the integration of IE and KDD for text mining can provide.

Traditional data mining assumes that the information to be "mined" is already in the form of a relational database. Unfortunately, for many applications, electronic information is only available in the form of free natural-language documents rather than structured databases. Since IE addresses the problem of transforming a corpus of textual documents into a more structured database, the database constructed by an IE module can be provided to the KDD module for further mining of knowledge as illustrated in Figure 1. Information extraction can play an obvious role in text mining as illustrated.

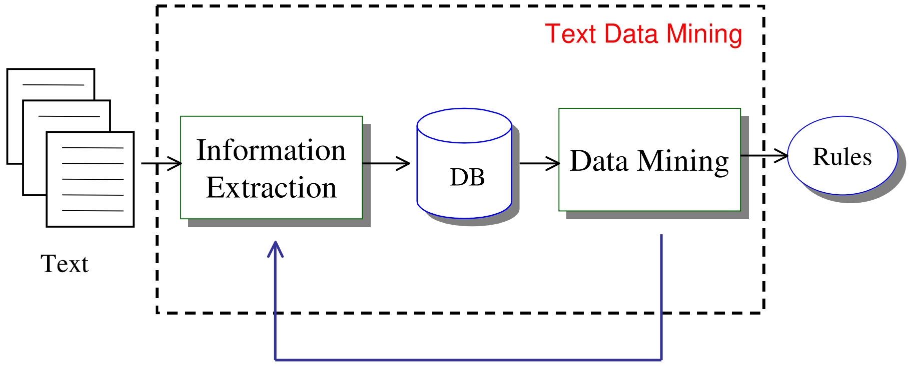

Figure 1: Overview of IE-based text mining framework

Although constructing an IE system is a difficult task, there has been significant recent progress in using machine learning methods to help automate the construction of IE systems [5, 7, 9, 23]. By manually annotating a small number of documents with the information to be extracted, a reasonably accurate IE system can be induced from this labeled corpus and then applied to a large corpus of text to construct a database. However, the accuracy of current IE systems is limited and therefore an automatically extracted database will inevitably contain significant numbers of errors. An important question is whether the knowledge discovered from this "noisy" database is significantly less reliable than knowledge discovered from a cleaner database. This paper presents experiments showing that rules discovered from an automatically extracted database are close in accuracy to that discovered from a manually constructed database.

A less obvious interaction is the benefit that KDD can in turn provide to IE. The predictive relationships between different slot fillers discovered by KDD can provide additional clues about what information should be extracted from a document. For example, suppose we discovered that computer-science jobs requiring "MySQL" skills are "database" jobs in many cases. If the IE system manages to locate "MySQL" in the `language` slot but failed to extract "database" in the `area` slot, we may want to assume there was an extraction error. Since typically the *recall* (percentage of correct slot fillers extracted) of an IE system is significantly lower than its *precision* (percentage of extracted slot fillers which are correct) [13], such predictive relationships might be productively used to improve recall by suggesting additional information to extract. This paper reports experiments in the computer-related job-posting domain demonstrating that predictive rules acquired by applying KDD to an extracted database can be used to improve the recall of information extraction.

The remainder of the paper is organized as follows. Section 2 presents some background information on text mining and IE. Section 3 describes a system called DiscoTEX (DISCOvery from Text EXtraction) that combines IE and KDD for text mining. Section 4 presents and discuss performance gains obtained in IE by exploiting mined prediction rules. Section 5 discusses some related work, Section 6 outlines directions for future research, and Section 7 presents our conclusions.

2 Background: Text Mining and Information Extraction

"Text mining" is used to describe the application of data mining techniques to automated discovery of useful or interesting knowledge from unstructured text [20]. Several techniques have been proposed for text mining including conceptual structure, association rule mining, episode rule mining, decision trees, and rule induction methods. In addition, Information Retrieval (IR) techniques have widely used the "bag-of-words" model [2] for tasks such as document matching, ranking, and clustering.

The related task of information extraction aims to find specific data in natural-language text. DARPA's Message Understanding Conferences (MUC) have concentrated on IE by evaluating the performance of participating IE systems based on blind test sets of text documents [13]. The data to be extracted is typically given by a template which specifies a list of slots to be filled with substrings taken from the document. Figure 2 shows a (shortened) document and its filled template for an information extraction task in the job-posting domain. This template includes slots that are filled by strings taken directly from the document. Several slots may have multiple fillers for the job-posting domain as in `programming languages`, `platforms`, `applications`, and `areas`.

We have developed machine learning techniques to automatically construct information extractors for job postings, such as those listed in the USENET newsgroup `misc.jobs.offered` [6]. By extracting information from a corpus of such textual job postings, a structured, searchable database of jobs can be automatically constructed; thus making the data in online text more easily accessible. IE has been shown to be useful in a variety of other applications, e.g. seminar announcements, restaurant guides, university web pages, apartment rental ads, and news articles on corporate acquisitions [5, 9, 23].

3 Integrating Data Mining and Information Extraction

In this section, we discuss the details of our proposed text mining framework, DiscoTEX (Discovery from Text EXtraction). We consider the task of first constructing a database by applying a learned information-extraction system to a corpus of natural-language documents. Then, we apply standard data-mining techniques to the extracted data, discovering knowledge that can be used for many tasks, including improving the accuracy of information extraction.

3.1 The DiscoTEX System

In the proposed framework for text mining, IE plays an important role by preprocessing a corpus of text documents in order to pass extracted items to the data mining module. In our implementations, we used two state-of-the-art systems for learning information extractors, Rapier (Robust Automated Production of Information Extraction Rules) [6] and BWI (Boosted Wrapper Induction) [15]. By training on a corpus of documents annotated with their filled templates, they acquire a knowledge base of extraction rules that can be tested on novel documents. Rapier and BWI

Document

```
Title: Web Development Engineer
Location: Beaverton, Oregon

This individual is responsible for design and implementation
of the web-interfacing components of the AccessBase server,
and general back-end development duties.

A successful candidate should have experience that includes:

   One or more of: Solaris, Linux, IBM AIX, plus Windows/NT
   Programming in C/C++, Java
   Database access and integration: Oracle, ODBC
   CGI and scripting: one or more of Javascript,
                      VBScript, Perl, PHP, ASP

Exposure to the following is a plus: JDBC, Flash/Shockwave,
FrontPage and/or Cold Fusion.

A BSCS and 2+ years experience (or equivalent) is required.
```

Filled Template

- title: "Web Development Engineer"
- location: "Beaverton, Oregon"
- languages: "C/C++", "Java", "Javascript", "VBScript", "Perl", "PHP", "ASP"
- platforms: "Solaris", "Linux", "IBM AIX", "Windows/NT"
- applications: "Oracle", "ODBC", "JDBC", "Flash/Shockwave", "FrontPage", "Cold Fusion"
- areas: "Database", "CGI", "scripting"
- degree required: "BSCS"
- years of experience: "2+ years"

Figure 2: Sample text and filled template for a job posting

Standard Term	Synonyms
"Access"	"MS Access", "Microsoft Access"
"ActiveX"	"Active X"
"AI"	"Aritificial Intelligence"
"Animation"	"GIF Animation", "GIF Optimization/Animation"
"Assembly"	"Assembler"
"ATM"	"ATM Svcs"
"C"	"ProC", "Objective C"
"C++"	"C ++", "C+ +"
"Client/Server"	"Client Server", "Client-Server", "Client / Server"
"Cobol"	"Cobol II", "Cobol/400", "Microfocus Cobol"
...	...

Table 1: Synonym dictionary (partially shown)

Job postings (600)

- Oracle $\in$ $application$ and QA Partner $\in$ $application$ $\rightarrow$ SQL $\in$ $language$
- HTML $\in$ $language$ and WindowsNT $\in$ $platform$ and Active Server Pages $\in$ $application$ $\rightarrow$ Database $\in$ $area$
- Java $\in$ $language$ and ActiveX $\in$ $area$ and Graphics $\in$ $area$ $\rightarrow$ Web $\in$ $area$
- UNIX $\notin$ $platform$ and Windows $\notin$ $platform$ and Games $\in$ $area$ $\rightarrow$ 3D $\in$ $area$
- AIX $\in$ $platform$ and Sybase $\notin$ $application$ and DB2 $\in$ $application$ $\rightarrow$ Lotus Notes $\in$ $application$
- C++ $\in$ $language$ and C $\in$ $language$ and CORBA $\in$ $application$ and Title=Software Engineer $\rightarrow$ Windows $\in$ $platform$

Figure 3: Sample mined prediction rules for computer-science jobs

have been demonstrated to perform well on realistic applications such as USENET job postings and seminar announcements.

After constructing an IE system that extracts the desired set of slots for a given application, a database can be constructed from a corpus of texts by applying the IE extraction patterns to each document to create a collection of structured records. Standard KDD techniques can then be applied to the resulting database to discover interesting relationships. Specifically, we induce rules for predicting each piece of information in each database field given all other information in a record. In order to discover prediction rules, we treat each slot-value pair in the extracted database as a distinct binary feature, such as "graphics $\in$ $area$", and learn rules for predicting each feature from all other features.

Similar slot fillers are first collapsed into a pre-determined standard term. For example, "Windows XP" is a popular filler for the `platforms` slot, but it often appears as "Win XP", "WinXP", 'MS Win XP", and so on. These terms are collapsed to unique slot values before rules are mined from the data. In our experiment, a manually-constructed synonym dictionary with 111 entries was employed. Table 1 shows the first 10 entries of the dictionary.

We have applied C4.5RULES [34] to discover interesting rules from the resulting binary data.

Resumé postings (600)

- HTML $\in$ $language$ **and** DHTML $\in$ $language$ $\rightarrow$ XML $\in$ $language$
- Illustrator $\in$ $application$ $\rightarrow$ Flash $\in$ $application$
- Dreamweaver 4 $\in$ $application$ **and** Web Design $\in$ $area$ $\rightarrow$ Photoshop 6 $\in$ $application$

- MS Excel $\in$ $application$ $\Rightarrow$ MS Access $\in$ $application$
- ODBC $\in$ $application$ $\Rightarrow$ JSP $\in$ $language$
- Perl $\in$ $language$ **and** HTML $\in$ $language$ $\Rightarrow$ Linux $\in$ $platform$

Figure 4: Sample rules of DISCOTEX for computer-science resumé postings

SF book descriptions (1,500)

- Sign of the Unicorn $\in$ $relatedbooks$ **and** American Science Fiction $\in$ $subject$ $\Rightarrow$ Knight of Shadows $\in$ $related\ books$
- Spider Robinson $\in$ $author$ $\Rightarrow$ Jeanne Robinson $\in$ $author$
- Roger Zelazny $\in$ $author$ $\Rightarrow$ 5 $\in$ $average\ rating$

Figure 5: Sample rules of DISCOTEX for book descriptions

Discovered knowledge describing the relationships between slot values is written in the form of production rules. If there is a tendency for "Web" to appear in the area slot when "Director" appears in the applications slot, this is represented by the production rule, "Director $\in$ $application$ $\rightarrow$ Web $\in$ $area$". Rules can also predict the absence of a filler in a slot; however, here we focus on rules predicting the presence of fillers. Sample rules mined from a database of 600 jobs extracted from the USENET newsgroup `austin.jobs` with RAPIER and C4.5RULES are shown in Figure 3.

We also applied RIPPER [10] and APRIORI [1] to discover interesting rules from the extracted data. APRIORI is a standard association rule mining algorithm which discovers all association rules that have support and confidence greater than the user-specified minimum support and minimum confidence. Sample rules mined from a database of 600 resumés extracted from the USENET newsgroup `misc.jobs.resumes` by BWI are shown in Figure 4. The first three rules are induced by RIPPER while the other three are found by APRIORI.

Since any IE or KDD module can be plugged into the DISCOTEX system, we also tested a highly-accurate information extractor (wrapper) manually developed for a book recommending system [28] to find interesting patterns from a corpus of book descriptions. Sample association rules mined from a collection of 1,500 science fiction (SF) book descriptions from the online `Amazon.com` bookstore are shown in Figure 5. Slots such as `authors`, `titles`, `subjects`, `related books`, and `average customer ratings` are identified from the corpus.

3.2 Evaluation

Discovered knowledge is only useful and informative if it is accurate. Therefore it is important to measure the accuracy of discovered knowledge on independent test data. The primary question we address in the experiments of this section is whether knowledge discovered from automatically extracted data (which may be quite noisy due to extraction errors) is relatively reliable compared to knowledge discovered from a manually constructed database.

For the dataset, 600 computer-science job postings to the newsgroup `austin.jobs` were collected and manually annotated with correct extraction templates. Ten-fold cross validation was used to generate training and test sets. RAPIER was used to learn the IE component and RIPPER was used as the KDD component. Rules were induced for predicting the fillers of the `languages`, `platforms`, `applications`, and `areas` slots, since these are usually filled with multiple discrete-valued fillers and have obvious potential relationships between their values (See [30] for more details on this experiment).

In order to test the accuracy of the discovered rules, they are used to predict the information in a database of user-labeled examples. For each test document, each possible slot-value is predicted to be present or absent given information on all of its other slot-values. Average performance across all features and all test examples were then computed.

The classification accuracy for predicting the absence or presence of slot fillers is not a particularly informative performance metric since high accuracy can be achieved by simply assuming every slot filler is absent. This is because the set of potential slot fillers is very large and only a small fraction of possible fillers is present in any given example. Therefore, we evaluate the performance of DISCOTEX using the IE performance metrics of precision, recall, and F-measure with regard to predicting slot fillers. These metrics are defined as follows:

$$precision = \frac{Number\ of\ actual\ slot\ values\ correctly\ predicted}{Number\ of\ slot\ values\ predicted\ to\ be\ present} \tag{1}$$

$$recall = \frac{Number\ of\ actual\ slot\ values\ correctly\ predicted}{Number\ of\ actual\ slot\ values} \tag{2}$$

We also report *F-measure* which is the harmonic mean of recall and precision:

$$F\!-\!measure = \frac{2 \times precision \times recall}{precision + recall} \tag{3}$$

Before constructing a database using an IE system, we filtered out irrelevant documents from the newsgroup using a bag-of-words Naive-Bayes text categorizer [26]. 200 positive documents (computer-science job postings) and 20 negative examples (spam postings, resumés, or non-cs job postings) are provided to the classifier for training. The performance of the classifier trained to predict the class "relevant" was reasonably good; precision is about 96% and recall is about 98%.

RAPIER was trained on only 60 labeled documents, at which point its accuracy at extracting information is somewhat limited; extraction precision is about 91.9% and extraction recall is about 52.4% . We purposely trained RAPIER on a relatively small corpus in order to demonstrate that labeling only a relatively small number of documents can result in a good set of extraction rules that is capable of building a database from which accurate knowledge can be discovered.

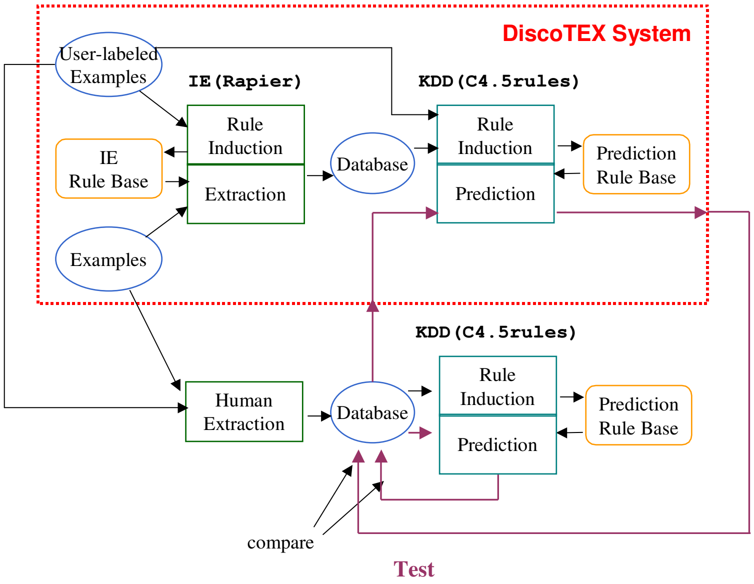

Figure 6: The system architecture - training and testing

Because of the two different training phases used in DISCOTEX, there is a question of whether or not the training set for IE should also be used to train the rule-miner. To clearly illustrate the difference between mining human-labeled and IE-labeled data, the IE training data are thrown away once they have been used to train RAPIER and ten-fold cross-validation is performed on the remaining 540 examples for evaluation of the data mining part. The same set of training examples was provided to both KDD systems, whereas the only difference between them is that the training data for DISCOTEX is automatically extracted by RAPIER after being trained on a disjoint set of 60 user-labeled examples. The overall architecture of the final system is shown in Figure 6.

Figure 7 shows the learning curves for precision, recall, and F-measure of both system as well as a random guessing strategy used as a baseline. The random guessing method predicts a slot-value based on its frequency of occurrence in the training data. Even with a small amount of user-labeled data, the results indicate that DISCOTEX achieves a performance fairly comparable to the rule-miner trained on a manually constructed database.

4 Using Mined Rules to Improve IE

After mining knowledge from extracted data, DISCOTEX can predict information missed by the previous extraction using discovered rules. In this section, we discuss how to use mined knowledge from extracted data to aid information extraction itself.

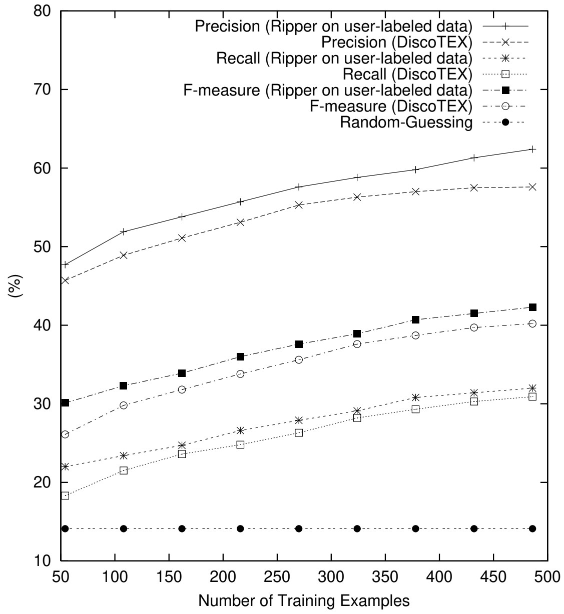

Figure 7: User-labeled data vs. IE-labeled data in rule accuracy

4.1 The Algorithm

Tests of IE systems usually consider two performance measures, *precision* and *recall* defined as:

$$precision = \frac{Number\ of\ correct\ fillers\ extracted}{Number\ of\ fillers\ extracted} \tag{4}$$

$$recall = \frac{Number\ of\ correct\ fillers\ extracted}{Number\ of\ fillers\ in\ correct\ templates} \tag{5}$$

Many extraction systems provide relatively high precision, but recall is typically much lower. Previous experiments in the job postings domain showed RAPIER's precision (e.g. low 90%'s) is singificantly higher than its recall (e.g. mid 60%'s) [6]. Currently, RAPIER's search focuses on finding high-precision rules and does not include a method for trading-off precision and recall. Although several methods have been developed for allowing a rule learner to trade-off precision and recall [11], this typically leaves the overall F-measure unchanged.

By using additional knowledge in the form of prediction rules mined from a larger set of data automatically extracted from additional unannotated text, it may be possible to improve recall without unduly sacrificing precision. For example, suppose we discover the rule "VoiceXML$\in$ $language$" $\rightarrow$ "Mobile $\in$ $area$". If the IE system extracted "VoiceXML $\in$ $language$" but failed to extract "Mobile $\in$ $area$", we may want to assume there was an extraction error and add "Mobile" to the area slot, potentially improving recall. Therefore, after applying extraction rules to a document, DISCOTEX applies its mined rules to the resulting initial data to predict additional potential extractions.

First, we show the pseudocode for the rule mining phase in Figure 8. A final step shown in the figure is filtering the discovered rules on both the training data and a disjoint set of labeled validation data in order to retain only the most accurate of the induced rules. Currently, rules

Input: $\mathcal{D}$ is the set of document.
Output: RB is the set of prediction rules.
Function RuleMining ($\mathcal{D}$)
Determine T, a threshold value for rule validation
Create a database of labeled examples (by applying IE to the document corpus, $\mathcal{D}$)
For each labeled example $D \in \mathcal{D}$ **do**
    F := set of slot fillers of D
    Convert F to binary features
Build a prediction rule base, RB (by applying rule miner to the binary data, F)
For each prediction rule $R \in RB$ **do**
    Verify R on training data and validation data
    If the accuracy of R is lower than T
        Delete R from RB
Return RB.

Figure 8: Algorithm specification: rule mining

Input: RB is the set of prediction rules.
    $\mathcal{D}$ is the set of documents.
Output: F is the set of slot fillers extracted.
Function InformationExtraction (RB, $\mathcal{D}$)
$F := \emptyset$.
For each example $D \in \mathcal{D}$ **do**
    Extract fillers from D using extraction rules and add them to F
    For each rule R in the prediction rule base RB **do**
        If R fires on the current extracted fillers
            If the predicted filler is a substring of D
                Extract the predicted filler and add it to F
Return F.

Figure 9: Algorithm specification: IE

that make *any* incorrect predictions on either the training or validation extracted templates are discarded. Since association rules are not intended to be used together as a set as classification rules are, we focus on mining prediction rules for this task.

The extraction algorithm which attempts to improve recall by using the mined rules is summarized in Figure 9. Note that the final decision whether or not to extract a predicted filler is based on whether the filler (or any of its synonyms) occurs in the document as a substring. If the filler is found in the text, the extractor considers its prediction confirmed and extracts the filler.

One final issue is the order in which prediction rules are applied. When there are interacting rules, such as "XML $\in$ $languages$ $\rightarrow$ Semantic Web $\in$ $areas$" and "Semantic Web $\notin$ $areas$ $\rightarrow$ .NET $\in$ $areas$", different rule-application orderings can produce different results. Without the first rule, a document with "XML $\in$ $language$" but without "Semantic Web $\in$ $area$" in its initial filled template will make the second rule fire and predict ".NET $\in$ $areas$". However, if the first rule is executed first and its prediction is confirmed, then "Semantic Web" will be extracted and the second rule can no longer fire. In DISCOTEX, all rules with negations in their antecedent conditions are applied first. This ordering strategy attempts to maximally increase recall by making as many confirmable predictions as possible.

To summarize, documents which the user has annotated with extracted information, as well as unsupervised data which has been processed by the initial IE system (which RAPIER has learned from the supervised data) are all used to create a database. The rule miner then processes this database to construct a knowledge base of rules for predicting slot values. These prediction rules are then used during testing to improve the recall of the existing IE system by proposing additional slot fillers whose presence in the document are confirmed before adding them to final extraction template.

4.2 Evaluation

To test the overall system, 600 hand-labeled computer-science job postings to the newsgroup `austin.jobs` were collected. 10-fold cross validation was used to generate training and test sets. In addition, 4,000 unannotated documents were collected as additional optional input to the text miner. Rules were induced for predicting the fillers of the `languages`, `platforms`, `applications`, and `areas` slots, since these are usually filled with multiple discrete-valued fillers and have obvious potential relationships between their values. Details of this experiment are described in [29].

Figure 10 shows the learning curves for recall and F-measure. Unlabeled examples are not employed in these results. In order to clearly illustrate the impact of the amount of training data for both extraction and prediction rule learning, the same set of annotated data was provided to both RAPIER and the rule miner. The results were statistically evaluated by a two-tailed, paired t-test. For each training set size, each pair of systems were compared to determine if their differences in recall and were statistically significant ($p < 0.05$).

DISCOTEX using prediction rules performs better than RAPIER. As hypothesized, DISCOTEX provides higher recall, and although it does decrease precision somewhat, overall F-measure is moderately increased. One interesting aspect is that DISCOTEX retains a fixed recall advantage over RAPIER as the size of the training set increases. This is probably due to the fact that the increased amount of data provided to the text miner also continues to improve the quality of the acquired prediction rules. Overall, these results demonstrate the role of data mining in improving the performance of IE.

Table 2 shows results on precision, recall and F-measure when additional unlabeled documents are used to construct a larger database prior to mining for prediction rules. The 540 labeled examples used to train the extractor were always provided to the rule miner, while the number of additional unsupervised examples were varied from 0 to 4,000. The results show that the more unsupervised data supplied for building the prediction rule base, the higher the recall and the overall F-measure. Although precision does suffer, the decrease is not as large as the increase in recall.

Although adding information extracted from unlabeled documents to the database may result in a larger database and therefore more good prediction rules, it may also result in noise in the database due to extraction errors and consequently cause some inaccurate prediction rules to be discovered as well. The average F-measure without prediction rules is 86.4%, but it goes up to 88.1% when DISCOTEX is provided with 540 labeled examples and 4,000 unlabeled examples. Unlabeled examples do not show as much power as labeled examples in producing good predic-

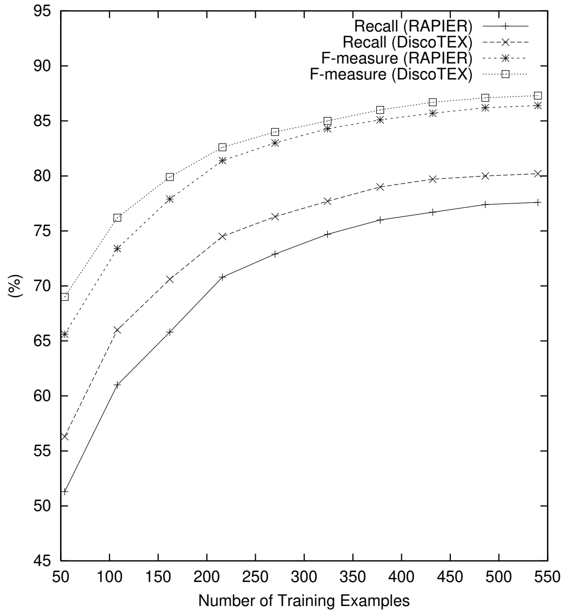

Figure 10: Recall and F-measures on job postings

Number of Examples for Rule Mining	Precision	Recall	F-Measure
0	97.4	77.6	86.4
540(Labeled)	95.8	80.2	87.3
540+1000(Unlabeled)	94.8	81.5	87.6
540+2000(Unlabeled)	94.5	81.8	87.7
540+3000(Unlabeled)	94.2	82.4	87.9
540+4000(Unlabeled)	93.5	83.3	88.1
Matching Fillers	59.4	94.9	73.1

Table 2: Performance results of DISCOTEX with unlabeled examples

tion rules, because only 540 labeled examples boost recall rate and F-measure more than 4,000 unlabeled examples. However, unlabeled examples are still helpful since recall and F-measure do slowly increase as more unlabeled examples are provided.

As a baseline, in the last row of Table 2, we also show the performance of a simple method for increasing recall by always extracting substrings that are known fillers for a particular slot. Whenever a known filler string, e.g. "C#", is contained in a test document, it is extracted as a filler for the corresponding slot, e.g. `language`. The reason why this works poorly is that a filler string contained in a job posting is not necessarily the correct filler for the corresponding slot. For instance, "HTML" can appear in a newsgroup posting, not in the list of required skills of that particular job announcement, but in the general instructions on submitting resumés.

5 Related Research

The most related system to our approach is probably DOCUMENT EXPLORER [14] which uses automatic term extraction for discovering new knowledge from texts. However, DOCUMENT EXPLORER assumes semi-structured documents such as SGML text unlike DISCOTEX developed for general natural-language text. Similarly, automatic text categorization has been used to map web documents to pre-defined concepts for further discovery of relationships among the identified concepts [24]. One of the limitations for these approaches is that they require a substantial amount of domain knowledge.

Several rule induction methods and association rule mining algorithms have been applied to databases of corporations or product reviews automatically extracted from the web [17, 16, 33]; however, the interaction between IE and rule mining has not been addressed. Recently a probabilistic framework for unifying information extraction and data mining has been proposed [25]. In this work, a graphical model using conditional probability theory is adopted for relational data, but experimental results on this approach are yet to be gathered. A boosted text classification system based on link analysis [12] is related to our work in spirit in that it also tries to improve the underlying learner by utilizing feedback from a KDD module.

6 Future Research

As mentioned in Section 3, DISCOTEX collapses similar slot-fillers in the extracted data into a canonical form based on a manually constructed dictionary. However, this approach has problems when a novel extracted entity is represented by similar but not identical strings in different documents. We have developed alternative rule-mining systems, TEXTRISE [31] and SOFTAPRIORI [32], that allow for partial matching of textual items based on a user-supplied similarity metric, such as edit-distance or bag-of-words cosine similarity. We are extending DISCOTEX to utilize those soft-matching rules to improve the underlying IE performance.

Good metrics for evaluating the interestingness of text-mined rules are also needed. One idea is to use a hierarchical lexical network to measure the semantic distance between the words in a rule, preferring "unexpected" rules where this distance is larger. We have developed such an approach using WordNet [3]. However, WordNet is general purpose and does not capture many of the semantic similarities in particular domains. Using a domain-specific taxonomy would be helpful for finding interesting rules. For example, this would allow ranking the rule "SQL Server $\rightarrow$ PHP" above "MySQL $\rightarrow$ PHP" since MySQL and PHP are both open source tools and therefore closer in a semantic hierarchy of software packages.

7 Conclusions

In this paper, we have presented an approach that uses an automatically learned IE system to extract a structured databases from a text corpus, and then mines this database with existing KDD tools. Our preliminary experimental results demonstrate that information extraction and data mining can be integrated for the mutual benefit of both tasks. IE enables the application of KDD to unstructured text corpora and KDD can discover predictive rules useful for improving IE performance.

This paper has presented initial results on integrating IE and KDD that demonstrate both of these advantages.

Text mining is a relatively new research area at the intersection of natural-language processing, machine learning, data mining, and information retrieval. By appropriately integrating techniques from each of these disciplines, useful new methods for discovering knowledge from large text corpora can be developed. In particular, the growing interaction between computational linguistics and machine learning [8] is critical to the development of effective text-mining systems.

Acknowledgements

This research was supported by the National Science Foundation under grant IIS-0117308.

References

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proceedings of the 20th International Conference on Very Large Databases (VLDB-94)*, pages 487–499, Santiago, Chile, Sept. 1994.

[2] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, New York, 1999.

[3] S. Basu, R. J. Mooney, K. V. Pasupuleti, and J. Ghosh. Evaluating the novelty of text-mined rules using lexical knowledge. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2001)*, pages 233–239, San Francisco, CA, 2001.

[4] M. W. Berry, editor. *Proceedings of the Third SIAM International Conference on Data Mining(SDM-2003) Workshop on Text Mining*, San Francisco, CA, May 2003.

[5] M. E. Califf, editor. *Papers from the Sixteenth National Conference on Artificial Intelligence (AAAI-99) Workshop on Machine Learning for Information Extraction*, Orlando, FL, 1999. AAAI Press.

[6] M. E. Califf and R. J. Mooney. Relational learning of pattern-match rules for information extraction. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, pages 328–334, Orlando, FL, July 1999.

[7] C. Cardie. Empirical methods in information extraction. *AI Magazine*, 18(4):65–79, 1997.

[8] C. Cardie and R. J. Mooney. Machine learning and natural language (Introduction to special issue on natural language learning). *Machine Learning*, 34:5–9, 1999.

[9] F. Ciravegna and N. Kushmerick, editors. *Papers from the 14th European Conference on Machine Learning(ECML-2003) and the 7th European Conference on Principles and Practice of Knowledge Discovery in Databases(PKDD-2003) Workshop on Adaptive Text Extraction and Mining*, Cavtat-Dubrovnik, Croatia, Sept. 2003.

[10] W. W. Cohen. Fast effective rule induction. In *Proceedings of the Twelfth International Conference on Machine Learning (ICML-95)*, pages 115–123, San Francisco, CA, 1995.

[11] W. W. Cohen. Learning to classify English text with ILP methods. In L. De Raedt, editor, *Advances in Inductive Logic Programming*, pages 124–143. IOS Press, Amsterdam, 1996.

[12] W. W. Cohen. Improving a page classifier with anchor extraction and link analysis. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 1481–1488, Cambridge, MA, 2003. MIT Press.

[13] DARPA, editor. *Proceedings of the Seventh Message Understanding Evaluation and Conference (MUC-98)*, Fairfax, VA, Apr. 1998. Morgan Kaufmann.

[14] R. Feldman, M. Fresko, H. Hirsh, Y. Aumann, O. Liphstat, Y. Schler, and M. Rajman. Knowledge management: A text mining approach. In U. Reimer, editor, *Proceedings of Second International Conference on Practical Aspects of Knowledge Management (PAKM-98)*, pages 9.1–9.10, Basel, Switzerland, Oct. 1998.

[15] D. Freitag and N. Kushmerick. Boosted wrapper induction. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, pages 577–583, Austin, TX, July 2000. AAAI Press / The MIT Press.

[16] R. Ghani and A. E. Fano. Using text mining to infer semantic attirbutes for retail data mining. In *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM-2002)*, pages 195–202, Maebash City, Japan, Dec. 2002.

[17] R. Ghani, R. Jones, D. Mladenić, K. Nigam, and S. Slattery. Data mining on symbolic knowledge extracted from the Web. In D. Mladenić, editor, *Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining (KDD-2000) Workshop on Text Mining*, pages 29–36, Boston, MA, Aug. 2000.

[18] M. Grobelnik, editor. *Proceedings of IEEE International Conference on Data Mining (ICDM-2001) Workshop on Text Mining (TextDM'2001)*, San Jose, CA, 2001.

[19] M. Grobelnik, editor. *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence(IJCAI-2003) Workshop on Text Mining and Link Analysis (TextLink-2003)*, Acapulco, Mexico, Aug. 2003.

[20] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, San Francisco, 2000.

[21] M. A. Hearst. Untangling text data mining. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL-99)*, pages 3–10, College Park, MD, June 1999.

[22] M. A. Hearst. What is text mining? http://www.sims.berkeley.edu/˜heast/text-mining.html, Oct. 2003.

[23] N. Kushmerick, editor. *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-2001) Workshop on Adaptive Text Extraction and Mining*, Seattle, WA, Aug. 2001. AAAI Press.

[24] S. Loh, L. K. Wives, and J. P. M. de Oliveira. Concept-based knowledge discovery in texts extracted from the Web. *SIGKDD Explorations*, 2(1):29–39, July 2000.

[25] A. McCallum and D. Jensen. A note on the unification of information extraction and data mining using conditional-probability, relational models. In *Proceedings of the IJCAI-2003 Workshop on Learning Statistical Models from Relational Data*, Acapulco, Mexico, Aug. 2003.

[26] A. McCallum and K. Nigam. A comparison of event models for naive Bayes text classification. In *Papers from the AAAI-98 Workshop on Text Categorization*, pages 41–48, Madison, WI, July 1998.

[27] D. Mladenić, editor. *Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining (KDD-2000) Workshop on Text Mining*, Boston, MA, Aug. 2000.

[28] R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the Fifth ACM Conference on Digital Libraries*, pages 195–204, San Antonio, TX, June 2000.

[29] U. Y. Nahm and R. J. Mooney. A mutually beneficial integration of data mining and information extraction. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, pages 627–632, Austin, TX, July 2000.

[30] U. Y. Nahm and R. J. Mooney. Using information extraction to aid the discovery of prediction rules from texts. In *Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining (KDD-2000) Workshop on Text Mining*, pages 51–58, Boston, MA, Aug. 2000.

[31] U. Y. Nahm and R. J. Mooney. Mining soft-matching rules from textual data. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-2001)*, pages 979–984, Seattle, WA, July 2001.

[32] U. Y. Nahm and R. J. Mooney. Mining soft-matching association rules. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management (CIKM-2002)*, pages 681–683, McLean, VA, Nov. 2002.

[33] J. M. Pierre. Mining knowledge from text collections using automatically generated metadata. In D. Karagiannis and U. Reimer, editors, *Proceedings of the Fourth International Conference on Practical Aspects of Knowledge Management (PAKM-2002)*, pages 537–548, Vienna, Austria, Dec. 2002. Springer. Lecture Notes in Computer Sicnece Vol. 2569.

[34] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo,CA, 1993.