Text Mining with Information Extraction

Un Yong Nahm and **Raymond J. Mooney**
Department of Computer Sciences,
University of Texas, Austin, TX 78712-1188
{pebronia,mooney}@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. However, this approach has problems when the same extracted entity or feature is represented by similar but not identical strings in different documents. Consequently, we also develop an alternate rule induction system called TextRISE, that allows for partial matching of textual items. Encouraging preliminary results are presented on applying these techniques to a corpus of Internet documents.

Introduction

The problem of *text mining*, i.e. discovering useful knowledge from unstructured text, is attracting increasing attention (Feldman 1999; Mladenić 2000; Grobelnik 2001). 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*. Information Extraction locates specific pieces of data from a corpus of natural-language texts. KDD considers the application of statistical and machine-learning methods to discover novel relationships in large relational databases. 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 unstructured 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.

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 (Califf 1999; Kushmerick 2001). By manually annotating a small number of documents with the information to be extracted, a fairly 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. In this paper we present experiments showing that rules discovered from an automatically extracted database is close in accuracy to that discovered from a manually constructed database.

Standard data mining methodologies are integrated with an IE component in the initial implementation of our framework. However, text strings in traditional databases often contain typos, misspellings, and non-standardized variations. The heterogeneity of textual databases causes a problem when we apply existing data mining techniques to text: the same or similar objects are often referred to using different textual strings. Consequently we propose a method, TextRISE for learning soft-matching rules from text using a modification of the RISE algorithm (Domingos 1996), a hybrid of rule-based and instance-based (nearest-neighbor) learning methods. This is good match for text mining since rule induction provides simple, interpretable rules, while nearest-neighbor provides soft matching based on a similarity metric.

The remainder of the paper is organized as follows. First we present background information on text data mining and information extraction. Second, we describe a system called DiscoTEX (DISCOvery from Text EXtraction) that combines IE and traditional KDD technologies to discover prediction rules from text. Next, we present TextRISE which allows partial matching in text mining. Finally we discuss related work, outline our future research plan, and present the conclusion.

Background

Text Mining

Text mining is defined as "the process of finding useful or interesting patterns, models, directions, trends, or rules from unstructured text". Several techniques have been proposed for text mining including conceptual structure, association rule mining, episode rule mining, decision trees, and rule induction methods so far. In addition, Information Retrieval (IR) techniques have been widely used (Baeza-Yates & Ribeiro-Neto 1999) for tasks such as document matching, ranking, and clustering, because a form of soft matching that utilizes word-frequency information typically gives superior results for most text processing problems (Salton 1989; Cohen 1998; Yang 1999).

Information Extraction

The proposed DiscoTEX framework considers information extraction (IE) as a key component for text mining. The goal of an information extraction system is to find specific data in natural-language text. DARPA's Message Understanding Conferences (MUC) has concentrated on IE by evaluating the performance of participating IE systems based on blind test sets of text documents (DARPA 1998). 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 1 shows a paired (shortened) document and template from an information extraction task in the resumé domain. This template includes only slots that are filled by strings taken directly from the document. Several slots may have multiple fillers for the resumé domain as in `(programming) languages`, `platforms`, and `areas`.

IE can be useful in a variety of applications, e.g. seminar announcements, course homepages, job postings and apartment rental ads. In particular, Califf (1998) suggested using machine learning techniques for extracting information from text documents in order to create easily searchable databases from the information, thus making the online text more easily accessible. For instance, information extracted from job postings in USENET newsgroup `misc.jobs.offered` can be used to build a searchable database of jobs. DiscoTEX is concerned with this aspect of IE, transforming unstructured texts to structured databases. Although most information extraction systems have been built entirely by hand until recently, automatic construction of complex IE systems began to be considered lately by many researchers. Recent proliferation of research on information extraction implies the possibility of using a successfully-built IE component for a larger text-mining system.

Document

```
I am a Windows NT software engineer seeking
a permanent position in a small quiet town
50 - 100 miles from New York City.

I have over nineteen years of experience in
all aspects of development of application
software, with recent focus on design and
implementation of systems involving multi-
threading, client/server architecture, and
anti-piracy. For the past five years, I have
implemented Windows NT services in Visual
C++ (in C and C++). I also have designed
and implemented multithreaded applications
in Java. Before working with Windows NT,
I programmed in C under OpenVMS for 5 years.
```

Filled Template

```
title: Windows NT software engineer
location: New York City
language: Visual C++, C, C++, Java
platform: Windows NT, OpenVMS
area: multi-threading, client/server,
      anti-piracy
years of experience: nineteen years
```

Figure 1: Sample message and filled template for a resumé posting

DiscoTEX: Integrating Data Mining and Information Extraction

In this section, we discuss the details of the proposed text mining framework. We consider the task of constructing a database with an information extraction system from the USENET newsgroup postings.

Information Extraction

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 information extraction systems, Rapier (Robust Automated Production of Information Extraction Rules) (Califf 1998) and BWI (Boosted Wrapper Induction) (Freitag & Kushmerick 2000). 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.

Rule Induction

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.

- 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 2: Sample rules of DiscoTEX for computer-science resumé postings

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.

We have applied Ripper (Cohen 1995) and C4.5rules (Quinlan 1993) to induce prediction rules from the resulting binary data. Ripper runs significantly faster than C4.5rules since it has an ability to handle set-valued features to avoid the step of explicitly translating slot fillers into a large number of binary features. A standard association rule mining algorithm was also applied to discover interesting associations between items. Sample rules mined from a database of 600 resumés BWI extracted from the USENET newsgroup `misc.jobs.resumes` are shown in Figure 2. The first three rules are induced by Ripper while the other three are from Apriori (Agrawal & Srikant 1994), a well-known association rule mining algorithm.

Experimental Results

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) is relatively reliable compared to knowledge discovered from a manually constructed database.

Experimental Methodology In order to test the accuracy of the discovered rules, they are used to predict the information in a disjoint 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 is then computed.

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 is used in the place of the IE part and Ripper for the KDD part. Rules were induced for predicting the fillers of the `(programming) languages`, `platforms`, `applications`, and `areas` slots, since these are usually filled with multiple discrete-valued fillers and have obvious potential relationships between their values.

The classification accuracy for predicting 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. It is because the set of potential slot fillers is very large and not fixed in advance, 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:

$$precision = \frac{\#ofPresentSlotValuesCorrectlyPredicted}{\#OfSlotValuesPredictedToBePresent} \quad (1)$$

$$recall = \frac{\#ofPresentSlotValuesCorrectlyPredicted}{\#OfPresentSlotValues} \quad (2)$$

F-measure is the harmonic mean of precision and recall and is computed as follows (when the same weight is given to precision and recall):

$$F\!-\!measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (3)$$

Results 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. 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.

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, we show a comparison with a disjoint IE training set. In this experiment, 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

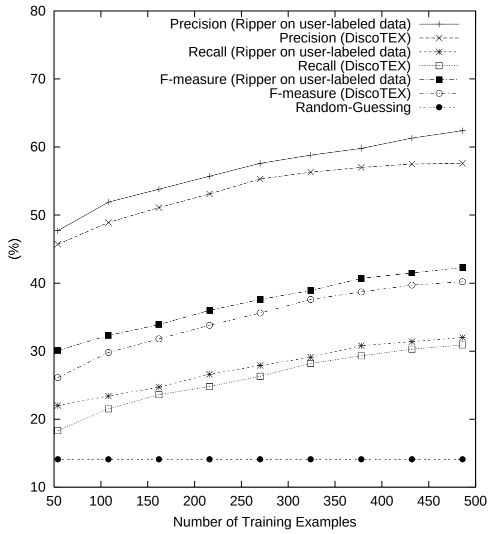

Figure 3: User-labeled data vs. IE-labeled data in rule accuracy

of 60 user-labeled examples. Figure 3 shows the learning curves for precision, recall, and F-measure of both system as well as the random guessing strategy. Random guessing method, predicting a slot-value based on its frequency of occurrence in the training data, is used as a benchmark to obtain non-trivial bounds on performance measures.

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. The recall curves in Figure 3 indicates that DiscoTEX does relatively worse with the first 60 training examples, but quickly improves with 60 additional examples. The results presented above employed 60 labeled examples to train the IE system. We also examined the effect of increasing the number of IE training examples to obtain a more accurate extraction module and found that if the training set for data mining to be automatically labeled by an IE module is large enough, DiscoTEX is able to achieve a fairly good performance with only a small amount of effort devoted to labeling IE training examples (See (Nahm & Mooney 2000) for more details).

TextRISE: Mining Soft-matching Rules from Text

One important task of text mining is the discovery of rules that relate specific words and phrases. Although existing methods for this task learn traditional logical rules, soft-matching methods that utilize word-frequency information generally work better for textual data. This section presents a rule induction system, TextRISE, that allows for partial matching of text-valued features by combining rule-based and instance-based learning. We present initial results applying TextRISE to a corpus of book descriptions and scientific papers retrieved from the web.

As mentioned previously, DiscoTEX manually collapsed similar slot-fillers in the extracted data into a canonical form. For example, "Win2k," "Windows 2000," and "Win 2000" are typical fillers extracted for the `platform` slot in the USENET resumé postings domain. All of those are mapped to a unique term by a synonym-checking dictionary before the rule mining step and treated as the same attribute afterwards. Instead of requiring or creating canonical slot-fillers that must match exactly, we developed an algorithm that allows partial matching of slot-fillers during the discovery process.

The RISE Algorithm

The RISE (Rule Induction from a Set of Exemplars) (Domingos 1996) algorithm is an integration of instance-based learning and rule induction. Unlike other combined model approaches, RISE is a unified single algorithm which is able to behave both as an instance-based classifier and a rule induction system. Instead of requiring rules to match exactly in order to make a prediction, RISE makes predictions by selecting the closest matching rule according to a standard distance metric typically used in nearest-neighbor methods. Flexible matching rules are acquired using a specific-to-general induction algorithm that starts with maximally specific rules for every example and then repeatedly minimally generalizes each rule to cover the nearest example it does not already cover, unless this results in a decrease in the performance of the overall rule base. Performance of a rule base is measured by conducting leave-one-out testing on the training data using the closest-matching rule for making predictions.

The TextRISE Algorithm

RISE is not directly applicable to mining rules from extracted text because: 1) its similarity metric is not text-based and 2) it learns rules for classification rather than text prediction. TextRISE addresses both of these issues.

Representation We represent an IE-processed document as a list of bags of words (BOWs), one bag for each slot filler. We currently eliminate 524 commonly-occurring stop-words but do not perform stemming. Figure 4 shows an example for a shortened research paper description and its BOW representation. Standard set-operations are extended to bags in the obvious way. A learned rule is represented as an antecedent that is a conjunction of BOWs for some subset of slots and a conclusion that is a predicted BOW for another slot (see Figure 6 for examples).

Like Whirl (Cohen 1998), TextRISE uses the standard cosine measure between two vectors for the similarity metric. The similarity of two slot-fillers is calculated by the *vector space model* (Baeza-Yates & Ribeiro-Neto 1999) for text. In this model, a document is repre-

Research Paper

Author: Raymond J. Mooney
Title: Integrating Abduction and Induction in Machine Learning
Abstract: This article discusses the integration of traditional abductive and inductive reasoning methods.
Bibliography: 1. Inductive Learning for Abductive Diagnosis
2. Abduction and Inductive Learning

Representation

Author	= {"raymond", "mooney"}
Title	= {"integrating", "abduction", "induction", "machine", "learning"}
Abstract	= {"article", "discusses", "integration" "traditional", "abductive", "inductive", "reasoning", "methods"}
Bibliography	= {"inductive" (2), "learning" (2), "abductive", "diagnosis", "abduction"}

Figure 4: An example of representation for a research paper

Input: ES is the training set.
Output: RS is the rule set.
Function TextRISE (ES)
$RS := ES$.
Compute $TextAcc(RS, ES)$.
Repeat
  For each rule $R \in RS$,
    Pick E' from ES that maximizes $Similarity(E', R)$
      (E' is not covered by R)
    If $TextAcc(RS, ES)$ is increased with replacement
      Replace R by $MostSpecificGeneralization(R, E')$
Until no increase in $TextAcc(RS, ES)$ is obtained.
Return RS.

Figure 5: The TextRISE rule-learning algorithm

sented as a vector of real numbers, where each component corresponds to a word and the component's value is its frequency in the document. The similarity of two documents x and y is the *cosine of the angle* between two vectors $\vec{x}$ and $\vec{y}$ representing x and y respectively.

Algorithm To extend the algorithm from classification to text prediction, we define a new measure for the accuracy of a rule set on an example set: $TextAcc(RS, ES)$ is the average cosine similarity of the predicted fillers for the examples in ES to the corresponding fillers predicted by a rule set RS. The algorithm for learning rules are described in Figure 5. The algorithm is a straightforward modification of RISE using the new similarity and predictive-accuracy metrics. Soft-matching rules are induced to predict the filler of each slot given the values of all other slots.

Bag intersection is used to compute the minimal generalization of two BOWs. The minimal generalization of two examples or rules is the minimal generalization of the BOWs in each of their corresponding slots. A rule is said to *cover* an example if all of its antecedents are

- **title** kate(1), cavanaugh(1), mystery(1), journal(1)
  reviews american(1), murder(1), characters(1)
  subject fiction(2), mystery(2)
  →
  author john(1), cathie(1)
- **synopses** players(1)
  subject children(1), juvenile(1), recreation(1), sports(1)
  →
  title baseball(1)
- **title** laboratory(1)
  subject science(2), nature(1)
  →
  reviews techniques(1), basic(1)
- **title** james(1), kelly(1) →
  comments science(1), fiction(1), alien(1), human(1), sf(1), nice(1)

Figure 6: Sample rules from book descriptions (3,000)

subbags of the example's corresponding BOWs. When finding the nearest example for generalization, we compute the similarity between each examples and the given rule to find an example with minimal distance to the rule. Our implementation of TextRISE makes use of the Bow library (McCallum 1996) for the bag-of-words text processing.

Rules Sample rules learned by TextRISE from the `Amazon.com` book descriptions are given in Figure 6. There are 6 slots in the book domain: `titles`, `authors`, `subject terms`, `synopses`, published `reviews`, and customer `comments`. Numbers associated to each word denote the number of occurrences in the bag.

Research paper data gathered by the Cora paper search engine (McCallum *et al.* 2000) are also employed. We used four slots, `author`, `title`, `abstract`, and `bibliography` from the data set. Figure 7 shows sample rules learned in this domain. Note that the third rule predicts bibliographic items instead of words in the bibliography section. This rule is generated by first mapping each cited paper to a unique token and applying the TextRISE algorithm on the consequent BOWs.

Finding Intra-slot Relationships Although all the rules shown so far are for prediction of one slot from all other slots, the learning algorithm of TextRISE is not restricted to this fixed form. Rules predicting two slots from all the other slots, e.g. "Organism in the book `title` implies life in `synopses` *and* biology in `subjects`.", can be mined without fundamental modifications to the algorithm by slightly changing the representation for rules. We also can learn rules involving intra-slot relationships by modifying the representation of examples and rules in TextRISE. Specifically, the definitions for a rule R and an example E can be modified by eliminating the consequent part. With this representation, all slots are predicted from all slots while

- **abstract** theory(1), decision(2), studied(1)
  bibliography based(1), learning(3), approach(1), domain(1), inductive(1), refinement(1)
  $\rightarrow$
  author sutton(1)
- **author** thorsten(1), joachims(1), dayne(1), freitag(1)
  abstract learning(1), web(1), implementation(1), engines(1)
  $\rightarrow$
  title machine(1), learning(1), web(1)
- **abstract** machine(1), learning(1), paper(1)
  $\rightarrow$
  bibliography Freund, Y. & Schapire, R. E., "A Decision-theoretic Generalization of On-line Learning and an Application to Boosting", 1996

Figure 7: Sample rules from CS research papers (1,000)

- **title** american(1)
  synopses native(1), american(1), present(1)
  subject children(1), native(1), american(1), north(1) america(1), indians(1)
- **title** space(1), camp(1)
  synopsesspace(1), shuttle(1), astronauts(1), program(1)
  reviews science(1), nasa(1)
  subject children(1), science(1), technology(1), nonfiction(1), juvenile(1)

Figure 8: Sample rules from children's book description (1,000)

only one slot is predicted from all the other slots in the original version. A rule covers an example if all of its slots are more general than the example's corresponding slots.

Rules produced by this algorithm specify relationships between terms implicitly. For instance, a rule of "`subject` = {"shakespeare", "classics", "literature"}" can be interepreted as ""shakespeare", "classics", and "literature" frequently occur together in the `subject` slot". This version of TEXTRISE works like clustering algorithms because the resulting rules are generalizations of rules that are close to each other. Each rule produced by this algorithm can be viewed as a *centroid* of a cluster consisting of all examples closer to that rule than any other rules. Sample rules produced from this version of TEXTRISE are presented in Figure 8.

It becomes clear that this version of TEXTRISE works like a clustering algorithm when we apply it to a collection of book descriptions documents with several genres. Many of the rules learned from the 6,000 book descriptions, 1,000 from each genre, put constraints only on the `subject` slot, e.g. "`subject` = {"children", "juvenile", "fiction"}", "`subject` = {"literature", "classics", "criticism"}", "`subject` = {"fiction", "mystery", "detective"}", or "`subject` = {"engineering", "science"}". In this case, the `subject` slot fillers can be viewed as pre-assigned labels for each cluster in the collection.

- **subject** physics (1), astronomy (1), science (2), nature (1), quality (1)
  $\rightarrow$
  synopses universe(1)
- **title** physicist (1)
  synopses science (2), nature(1)
  $\rightarrow$
  subject science (1)

Figure 9: Sample rules from science-book description (1,000)

Semantic Rule Expansion A potentially useful change to the generalization algorithm is to use a semantic hierarchy such as WordNet (Fellbaum 1998). For example, the terms "thermodynamics" and "optics" could be generalized to "physics" using the *hypernyms* hierarchy of WordNet. Similarly, the two rules "thermodynamics $\in$ *subject* $\rightarrow$ heat, modern, theory, waves $\in$ *synopses*" and "optics $\in$ *subject* $\rightarrow$ electromagnetics, laser, waves $\in$ *synopses*" might be minimally generalized to the rule "physics $\in$ *subject* $\rightarrow$ waves $\in$ *synopses*" if a semantic lexicon provided by WordNet is utilized. "Thermodynamics" and "optics" have the common parent, "physics", in the hypernym tree.

The current implementation of TEXTRISE generates an empty filler slot for the `subject` slot in this case because it is not able to recognize the semantic relationships between the two slot fillers. This change would requires a redefinition of distance between words in terms of the WordNet hierarchy; however, we implemented a simpler version of semantic TEXTRISE by expanding each example with its semantic parents in WordNet. In the initial stage, every BOW for an example is augmented by taking semantic parents of each word in that BOW. For simplicity, we did not take word sense disambiguation problem into account and considered only the primary meaning of each word. Figure 9 shows rules generated by applying this version of TEXTRISE on the science-book domain. "Physics" in the first rule shown in Figure 9 is an example of a word not originally from a document but expanded from another word. "Physicist" in the second rule is also a generalization of its child nodes in the Wordnet, such as "einstein".

Experimental Results

We evaluate the performance of TEXTRISE by comparing it with a standard nearest-neighbor prediction strategy. Detailed description on this experiment can be found in (Nahm & Mooney 2001).

Experimental Methodology The `Amazon.com` book descriptions domain is employed in our evaluation of TEXTRISE. We manually developed simple pattern-based IE systems or "wrappers" to automatically extract various labeled text-fields from the original HTML

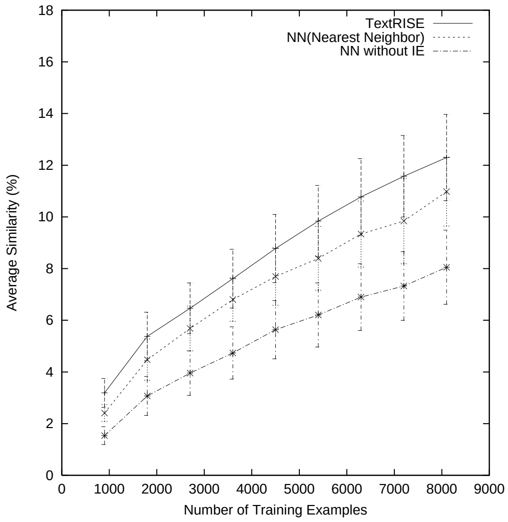

Figure 10: Average similarities for book data

documents. The text extracted for each slot is then processed into a bag-of-words after removal of stop-words and remaining HTML commands. The book data set is composed of 6 subsets, science fiction, literary fiction, mystery, romance, science, and children's books. 1,500 titles were randomly selected for each genre.

Unlike a standard rule learner that predicts the presence or absence of a specific slot value, TEXTRISE predicts a bag-of-words for each slot. Therefore, we evaluate the performance of TEXTRISE by measuring the average cosine similarity of the predicted slot values to the actual fillers for each slot. We compare the system to a standard nearest-neighbor method. In both methods, prediction is made by selecting the closest rule/example using only the text in the antecedent slots. We also tested nearest-neighbor without using information extraction to show the benefit of IE-based text mining. The experiments were performed on the 9,000 book descriptions using ten-fold cross validation.

Results Learning curves for predicting the `title` slot are shown in Figure 10. The graph shows 95% confidence intervals for each point. All the results were statistically evaluated by a one-tailed, paired t-test ($p < 0.05$). The results indicate that TEXTRISE does best, while nearest-neighbor without IE does worst. This shows TEXTRISE successfully summarizes the input data in the form of prediction rules. We conducted the same experiments for other slots, and found similar results except for predicting the `author` slot.

Related Research

The most relevant systems to our approach might be KDT (Knowledge Discovery in Textual Databases) (Feldman & Dagan 1995) and FACT (Feldman & Hirsh 1996). Both systems allude to the use of IE in text mining; however, they use texts manually tagged with a limited number of fixed category labels instead of actually using automated text categorization. Recently an extension of KDT for web mining is suggested by discovering conceptual knowledge from web documents using automated text categorization (Loh, Wives, & de Oliveira 2000). One of the limitations for these approaches is that they require a substantial amount of domain knowledge. Ghani *et al.* (2000) applied several rule induction methods to a database of corporations automatically extracted from the web using rule induction system such as C5.0 and FOIL; however, all induced rules are hard-matching rules that must exactly match extracted text.

Future Research

Currently the main drawback of TEXTRISE in terms of the data-mining perspective is its quadratic time complexity for rule discovery. TEXTRISE is now strictly based on the RISE algorithm, requiring time $O(e^2, a^2)$ where e is the number of examples, and a the number of attributes. We plan to address the efficiency issue by modifying the rule-learning algorithm of TEXTRISE, e.g. speeding up the nearest-neighbor search using heuristics, randomly picking rules for generalization, or introducing various sampling techniques.

Good metrics for evaluating the interestingness of text-mined rules are clearly needed. One idea is to use a hierarchical network to measure the semantic distance between the words in a rule, preferring "surprising" rules where this distance is larger. Such an algorithm using WordNet has been proposed (Basu *et al.* 2001); however, WordNet is too general and huge a hierarchy to be useful in every specific domain. Using a smaller domain-specific taxonomy would be helpful for finding more interesting rules. For example, this would allow ranking the rule "beer → diapers" above "beer → pretzels" since beer and pretzels are both food products and therefore closer in the product hierarchy

Alternative metrics for measuring textual similarity will also be explored. For short extracted strings, string edit distance might be a more useful measure of textual similarity than the cosine measure.

Conclusions

In this paper, we showed that text-mining systems can be developed relatively rapidly and evaluated easily on existing IE corpora by utilizing existing Information Extraction (IE) and data mining technology. We presented an approach of using an automatically learned IE system to extract a structured databases from a text corpus, and then mining this database with traditional KDD tools. Our preliminary experimental results demonstrate that the knowledge discovered from such an automatically extracted database is close in accuracy to the knowledge discovered from a manually constructed database. Due to variability and diversity in natural-language data, some form of soft matching based on textual similarity is needed when discovering rules from text. We have also presented a hybrid system of rule-based and instance-based learning methods to discover soft-matching rules from textual databases automatically constructed via information extraction. We empirically showed how this approach can induce accurate predictive rules despite the heterogeneity of automatically extracted textual databases.

Acknowledgements

This research was supported by the National Science Foundation under grant IRI-9704943.

References

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

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

Basu, S.; Mooney, R. J.; Pasupuleti, K. V.; and Ghosh, J. 2001. 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*, 233–239.

Califf, M. E. 1998. *Relational Learning Techniques for Natural Language Information Extraction.* Ph.D. Dissertation, Department of Computer Sciences, University of Texas, Austin, TX. Also appears as Artificial Intelligence Laboratory Technical Report AI 98-276.

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

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

Cohen, W. W. 1998. Providing database-like access to the web using queries based on textual similarity. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data (SIGMOD-98)*, 558–560.

DARPA., ed. 1998. *Proceedings of the Seventh Message Understanding Evaluation and Conference (MUC-98).* Fairfax, VA: Morgan Kaufmann.

Domingos, P. 1996. Unifying instance-based and rule-based induction. *Machine Learning* 24:141–168.

Feldman, R., and Dagan, I. 1995. Knowledge discovery in textual databases (KDT). In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95)*, 112–117.

Feldman, R., and Hirsh, H. 1996. Mining associations in text in the presence of background knowledge. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, 343–346.

Feldman, R., ed. 1999. *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99) Workshop on Text Mining: Foundations, Techniques and Applications.*

Fellbaum, C. D. 1998. *WordNet: An Electronic Lexical Database.* Cambridge, MA: MIT Press.

Freitag, D., and Kushmerick, N. 2000. Boosted wrapper induction. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, 577–583.

Ghani, R.; Jones, R.; Mladenić, D.; Nigam, K.; and Slattery, S. 2000. Data mining on symbolic knowledge extracted from the web. In Mladenić, D., ed., *Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining (KDD-2000) Workshop on Text Mining*, 29–36.

Grobelnik, M., ed. 2001. *Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM-2001) Workshop on Text Mining.* To be published.

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

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

McCallum, A. K.; Nigam, K.; Rennie, J.; and Seymore, K. 2000. Automating the construction of internet portals with machine learning. *Information Retrieval* 3(2):127–163. www.research.whizbang.com/data.

McCallum, A. K. 1996. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/~mccallum/bow.

Mladenić, D., ed. 2000. *Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining (KDD-2000) Workshop on Text Mining.*

Nahm, U. Y., and Mooney, R. J. 2000. 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*, 51–58.

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

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

Salton, G. 1989. *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer.* Addison-Wesley.

Yang, Y. 1999. An evaluation of statistical approaches to text categorization. *Journal of Information Retrieval* 1(1/2):67–88.