Semantic Bootstrapping with a Cluster-Based Extension to DIPRE

Dmitri Bobrovnikoff
dmitrib@cs.stanford.edu

June 1, 2000

Abstract

The practical applications of information extraction are currently limited by the need to hand-construct search patterns and lexicons and / or to have available large labelled training sets. To address this issue, we present a semantic bootstrapping technique based on Brin's DIPRE algorithm. The basic algorithm is extended by using clustering to group similar occurrences when extracting new patterns.

1 Introduction

Information extraction is a technique for extracting narrowly focused target relations from large information bases such as news wire archives or the World Wide Web. Unfortunately, most current information extraction systems are quite labor-intensive; they may require hand-coded search patterns, data massaging specific to the desired target relation, and often, perhaps most significantly, large numbers of labelled training examples.

One approach to automating the information extraction process is a class of techniques called semantic bootstrapping. In general, these techniques use some form of machine learning to recognize semantic patterns in large amounts of unlabelled data. The recognized patterns are then applied iteratively to extract more semantics. Effectively, semantic bootstrapping applies semantic labels to the data based on what it already knows, then labels even more data based on the new labels it applied.

The DIPRE (Dual Iterative Pattern Relation Extraction) algorithm, described in [Brin 98], is a semantic bootstrapping algorithm oriented toward extracting simple relations from the Web, starting with only a few seed relations. Based on these initial relations, the algorithm automatically recognizes new patterns. The patterns are then used to extract new relations. This process is repeated iteratively, extracting more relations and generating more patterns on each pass until a threshold is reached, at which point the algorithm terminates.

This paper presents an attempt to address one of DIPRE's weaknesses by extending it with a clustering algorithm that is used to group occurrences.

In addition to the clustering extension, one of the primary differences between this experiment and the one described in [Brin 98] is the nature of the data on which information extraction / semantic bootstrapping was performed. Brin's work was focused on extracting (*author*, *title*) pairs from Web pages. As one might expect (and as is entirely reasonable, given the nature of the data), the automatically generated patterns made heavy use of HTML formatting tags to find authors and titles on various book-related sites, often in tables or lists. One factor that motivated this choice of project was the desire to see whether DIPRE could handle more free-form natural language text such as news wire.

The paper proceeds as follows. Section 2 defines the problem. The DIPRE algorithm and my extensions to it are presented in Section 3, and the implementation is described in Section 4. Section 5 presents the results obtained. Related work is touched up on in Section 6 and results are discussed in Section 7.

2 Extracting Relations Using Semantic Bootstrapping

2.1 General Description of Problem

The system described in this paper attempts extract a set of tuples R representing instances of a particular well-defined relation from arbitrary, unlabelled English sentences. It starts with a small seed set of examples of the desired relation, which it analyzes and uses to extract further instances of the relation.

2.2 Acquisitions and Profits

To explore the strengths and weaknesses of the algorithm, it was applied to two distinct target relations: company acquisitions of the form (*acquirer*, *acquiree*) and profit announcements of the form (*company*, *profits*). The acquisition example will be used as an illustration throughout this paper.

The source data was a large amount of Wall Street Journal news wire from 1988. These example were chosen because they are simple, well-defined and in abundant supply in the data set. In principle, this algorithm should work with any relatively small relation for which a large amount of data is available.

3 Description of Algorithm

3.1 Overview of DIPRE

The core of this project was an independent implementation of DIPRE based on Brin's description in [Brin 98]; see that reference for a more in-depth treatment of the specifics of the algorithm. A number of minor modifications to the original algorithm were necessary due to differences in the data sets. The basic algorithm, as implemented, is as follows:

1. Extract sentences from raw data based on SGML tags. Filter the sentences so only those containing potential *acquirer* / *acquiree* company names remain.

2. Initialize the relation set to a small set of seed relations: $R = R_{\text{seed}}$. Each relation $r \in R$ is of the form (*acquirer*, *acquiree*).

3. Find all occurrences of each relation $r \in R$ in the data set, yielding an occurrence set O. Each occurrence $o \in O$ is of the form (*prefix*, *acquirer*, *middle*, *acquiree*, *suffix*). The occurrences capture some of the context surrounding the *acquirer* / *acquiree*, to be used in the pattern generation step. Specifically, the context is defined by the Perl regular expressions:

   - *prefix*: `.{0,10}`
   - *middle*: `.*?`
   - *suffix*: `.{0,10}`

4. Extract patterns based on the occurrence set O. This yields a pattern set P. Each pattern $p \in P$ is of the form (*prefix*, *middle*, *suffix*), where *prefix*, *middle*, and *suffix* are all strings.

   (a) Set the pattern set P to empty.

   (b) Group occurrences into a set of occurrence groups G, based on the *middle* attribute of each occurrence.

   (c) For each group $g \in G$ such that $|g| \geq n$ (n was at least 2 in our experiments),

i. Generate a pattern p such that $middle = g.middle$ (note that all members of g must have the same $middle$), *prefix* is the longest common suffix of all *prefix*es of the occurrences in the group and *suffix* is the longest common prefix of all *suffix*es of the occurrences in the group.

ii. Calculate the specificity $specificity = \mid p.prefix \mid \cdot \mid p.middle \mid \cdot \mid p.suffix \mid$.

iii. If *specificity* is above a threshold t, add p to P. Otherwise, use a partitioning algorithm to create sub-groups if possible ($|g| \geq n$ still must hold for sub-groups) and return to the pattern generation step.

5. Apply the pattern set P to the data set to extract a new relation set R. A Perl regular expression of the form *prefix* <*proper-name-pattern*> *middle* <*proper-name-pattern*> *suffix* (for the acquisitions data set – <*proper-name-pattern*> would get replaced by expressions appropriate to the application for other data sets) is built from each pattern p. All sentences in the data set are matched against these expressions and matching relations are extracted.

6. If the maximum number of iterations has not been exceeded, start the next iteration at step 3.

3.2 Extensions to DIPRE

One of the weaker areas of the original algorithm was the method of subdividing the patterns if the specificity requirements were not met. Brin's algorithm simply looked at the character following the *urlprefix* (not applicable for the Wall Street Journal data set) and created subgroups on the basis of that character.

In an effort to extend the algorithm, I implemented a number of different partitioning algorithms:

- One similar to Brin's that looks at the penultimate prefix character (the final prefix character is almost always a space for this particular application) and creates new groups based on that character.
- An algorithm that recursively divides the group in half until either the specificity requirement is met or the sub-group size becomes too small,
- A division scheme that chooses random subsets of groups.
- A clustering algorithm based on sentence similarity. KNN with $k = \frac{1}{2}|g|$, where g is the group to split, is used. The first occurrence in the group is arbitrarily chosen as the central cluster location. Half the occurrences are clustered around that first occurrence to form one group and the rest of the occurrences form a second group. The matching coefficient is used as a measure of similarity. If the specificity of the sub-groups still doesn't meet the requirements, the clustering algorithm can be called recursively to further sub-divide them.

4 Implementation

4.1 Platform

The system described in this paper was implemented on the Sun workstations in Sweet Hall. The main algorithm was a Perl program. For speed, the routines for filtering out the initial data set made use of several standard UNIX command-line tools, namely *grep* and *sed*.

4.2 Performance Issues

Given the size of the data sets to which information extraction techniques are generally applied (for the Wall Street journal news wire, more than 150 megabytes; Web archives can reach into many gigabytes or even terabytes), performance is definitely a factor that needs to be considered.

This implementation pre-filtered the data, extracting sentences, removing SGML tags and removing sentences that did not have any patterns that could be potentially classified as company names. The main algorithm was then run using this pre-filtered data so the filtering stage did not have to be performed each time.

An initial performance problem was due to attempting to load all sentences into memory initially; this slowed things down considerably because there was insufficient physically memory available to contain the entire data set and a large amount of disk swapping was incurred while iterating over the list of sentences. The problem was solved by reading the filtered data file one line at a time; while this approach required repeated accesses to the same file locations, it worked better in practice than the disk swapping.

Generally, the most efficient available data structures (*e.g.*, hash tables) were used in as many places as possible. The DIPRE algorithm itself was designed specifically to work with large amounts of data (and takes a number of shortcuts in order to do so). Even so, the algorithm took a minimum of several minutes per iteration for a 50 MB data set and slowed down even more as it added more patterns.

5 Experiments

5.1 Testing

Different versions of DIPRE were run on the two data sets (profits and acquisitions) described in Section 2. The performance of the algorithm was monitored while it was running. "Improvements" that hurt performance were usually obvious early on and could be corrected by aborting the process; this was necessary, since the algorithm may take many hours if it finds large numbers of tuples.

Once the run had finished, a sample of the generated tuples was selected and scored by reading the original sentences in the WSJ and determining whether the intended meaning matched the definition of the relation. Since both relations are quite simple and well-defined, there was little ambiguity in the scoring.

5.2 Data

Table 1 shows all tuples obtained from the profits relation at iteration 10.

Table 2 shows a small sample of the 602 tuples found for the acquisitions relation by iteration 2.

6 Related Work

The specific applications of semantic bootstrapping vary. However, different approaches share a common goal of learning some sort of semantic relationship from a large amount of unlabelled data.

[Riloff 99] presents a system for bootstrapping lexicons for use in a separate information extraction system. To increase the reliability of the generated lexicons, a multi-level approach is used. This approach combines a low-level bootstrapping procedure ("mutual bootstrapping") with a high-level procedure ("meta bootstrapping") that takes only the best results of the low-level procedure.

In [Jones 99], the multi-level approach just described is applied to a text classification task.

[Roark 98] describes the use of noun phrase co-occurrence bigrams for lexicon construction.

Finally, [Brin 98], the work on which this project is based, extracts (*author*, *title*) relations from web pages using the DIPRE algorithm.

acquiree	acquirer	score
McDonnell Douglas	$26 million	+
United Technologies	$33.3 million	+
Time	$15.3 million	+
United Technologies	$41.4 million	+
McDonnell Douglas	$21 million	+
McDonnell Douglas	$40 million	+
Atari	$12.1 million	+
Control Data	$10.8 million	+
Sun	$20.6 million	+
Bell Atlantic	$1.24 billion	+
Zayre	$78.2 million	+
Dow Chemical	$312 million	+
Morino Associates	$6.2 million	+
Tektronix	$18.7 million	+
Homestake	$146.4 million	+

Table 1: Sample tuples found for the profits relation, with scores. The data set was 100 MB of WSJ news wire from 1988; the specificity threshold was 50 and the subdivision algorithm was recursive halving. This sample was taken from the final output of the tenth iteration.

7 Discussion

7.1 Results

For information extraction tasks, high precision is usually more desirable than high recall. Because there is so much information available, the primary task of information extraction is to present the pieces that are actually of interest while excluding the flotsam; missing even a fairly large chunk of the total data is often OK if it means that that returned data is of high quality. Unfortunately, large, unlabelled data sets by nature do not lend themselves easily to exact calculations of precision and recall. While I have been able to make some estimates of the accuracy of my algorithm by hand-scoring its results, estimating coverage is more problematic. Counting the number of false negatives in a large, unlabelled data set is simply impractical. From inspections of the data, it is obvious that the algorithm does not succeed in extracting every tuple that in fact meets the definition of the relation. As a very *ad hoc* estimate, I would say that recall is perhaps 30%. However, as discussed above, this is perfectly acceptable for many IE applications.

For the profits data set, a hand check of all 15 tuples returned after 10 iterations yielded a precision of 100%. However, this score is somewhat misleading. For this particular example, recall was very low – a quick inspection reveals that there are many more than 15 profit announcements in over 100 MB of Wall Street Journal news wire. Another factor that makes the precision less impressive than it appears is the the seed set contained several of the companies that appear in the list. While this is not a surprising fact, it does skew the results toward artificially high precision. The basic problem is that DIPRE appears to be fairly sensitive to the initial set of seed tuples which it is given. I tried several different combinations of seed tuples for this data set, but was never able to get more than about 15 good tuples as a result. The initial tuples would generate 2 or 3 patterns, then the algorithm would rapidly reach a steady state in which no new patterns were generated. I only experimented with around 5 seed tuples, which Brin found to be sufficient, and which was sufficient for the other data set; perhaps a larger and more diverse set of patterns is required to get the bootstrapping off to a good start.

The acquisitions data set suffered from exactly the opposite problem: so many patterns got generated

acquiree	acquirer	score
Paramount	Gulf	+
Scrimgeour Vickers Taiwan Ltd.	Citicorp	+
Shell	Royal Dutch	+
R.J. Reynolds	RJR Nabisco Inc.	+
USF&G Financial Services Corp.	USF&G Corp.	+
Jarman Shoe Cos.	Genesco Inc.	+
GTE Florida	GTE Corp.	+
Asia	Drexel Burnham Lambert Inc.	-
CSX Transportation Inc.	CSX Corp.	+
Magnavox Electronic Systems Co.	North American Philips Corp.	+
Lockheed Aeronautical Systems	Lockheed Corp.	+
Rubber Co.	Bridgestone Corp.	+
Qintex	Qintex Ltd.	+
BT Telecom Inc.	Base Ten Systems Inc.	+
Lyondell Petrochemical Co.	Atlantic Richfield Co.	+
Canada Ltd.	General Motors Corp.	-
Continental Airlines	Texas Air Corp.	+
Mission Energy Co.	SCEcorp	+
NBC	General Electric Corp.	+
Bankers Trust Australia Ltd.	Bankers Trust New York	+

Table 2: Sample tuples found for the acquisitions relation, with scores. The data set was 50 MB of WSJ news wire from 1988; the specificity threshold was 2 and the subdivision algorithm was recursive halving. This sample was taken from the final output of the second iteration.

that the algorithm soon bogged down as it slogged through them. A few of the initial patterns extracted a large number of new relations, which then generated new patterns etc. After about 4 iterations, so many new patterns had been generated that spurious patterns began to creep in and taint the results; the algorithm eventually began to emit totally bogus tuples such as ("But Monday", "Rospatch"). However, prior to the destabilization, surprisingly good results were obtained. After iteration 2, 20 randomly selected tuples from a total of 602 were checked. Of these, 18 were true positives for a precision of 90%. The two errors were both relatively uninteresting, byproducts of the manner in which the regular expressions recognize company names.

7.2 Utility of the Clustering Extension

I originally focused on extending the performance of the grouping mechanism because it seemed to be one of the weaker and more *ad hoc* portions of the DIPRE algorithm. However, none of the four partitioning algorithms, including clustering, that I implemented had a discernible effect on the patterns and tuples generated. This was a definite disappointment. The algorithm does require some form of partitioning, especially when the sensitivity threshold is set high; however, it doesn't seem very picky about what kind. Possibly the effects of the different partitioning algorithms could be better seen with differently structured data sets than the ones I used.

7.3 Problems with DIPRE

As demonstrated in [Brin 98] and here, DIPRE is quite productive and fairly accurate given a sufficiently large data set. This holds for different domains, even with relatively free-form natural language text such as news wire reports. However, the algorithm is far from perfect.

Most notably, it seems unlikely that it can be easily scaled effectively tackle more complex relations. While a straightforward extension of the current two-slot pattern matching technique to three or more slots is possible, such an approach would result in a combinatorial explosion of possible orderings. Also, data would likely become sparse with great rapidity as the number of slots increases.

Another problem is the simplicity of the chunking method used. The regular expressions generated by the ad hoc pattern generator are simple strings that don't fully utilize the expressive power of general regular expressions. However, even if richer regular expressions were generated, they still wouldn't be able to capture the relations between words in a sentence with much precision.

Finally, the system eventually becomes unstable after a number of iterations. This is because the occasional but inevitable introduction of incorrect relations causes a snowball effect, leading to bad patterns and the introduction of more bad relations. The instability problem is common to most bootstrapping algorithms and has been reported previously, *e.g.*, in [Riloff 99].

7.4 Goals

There were two basic goals for this project:

- To determine whether DIPRE is capable of extracting useful relations from relatively free-form news wire text in addition to the more structured occurrences investigated by Brin using a Web data set.
- To extend DIPRE by using clustering to group occurrences and evaluate the utility of this extension.

These goals have been met, as described in the above sections.

8 Acknowledgements

This project was largely based on Brin's description of the DIPRE algorithm in [Brin 98]. However, my implementation of the algorithm was completely from scratch.

Portions of the clustering code were adapted from my implementation of Assignment 3.

References

[Brin 98] Brin, S. Extracting Patterns and Relations from the World Wide Web. WebDB Workshop at EDBT '98. 1998.

[Jones 99] Jones, R., McCallum, A., Nigam, K. and E. Riloff. Bootstrapping for Text Learning Tasks. In *IJCAI-99 Workshop on Text Mining: Foundations, Techniques and Applications*. 1999.

[Manning 00] Manning, C. and Schütze, H. *Foundations of Statistical Natural Language Processing*. MIT Press. Cambridge, Massachusetts. 2000.

[Riloff 99] Riloff, E. and Jones, R. Learning Dictionaries for Information Extraction Using Multi-level Boot-strapping. In *AAAI*. 1999.

[Roark 98] Roark, B. and Charniak, E. Noun-Phrase Co-Occurence Statistics for Semi-Automatic Semantic Lexicon Construction. *COLING-ACL*: 1110-1116. 1998.