Discovery of frames and relationship from natural language text

Chakrabarti P.* and Basu J.K.
*Sir Padampat Singhania University, Udaipur-313601, Rajasthan, India, prasun9999@rediffmail.com
Heritage Institute of Technology, Kokata-700107, West Bengal, India, basu.jayanta@yahoo.co.in

Abstract- Purpose of this paper is to addresses the problem of information discovery in large collections of text. For users, one of the key problems in working with such collections is determining where to focus their attention. Text documents often contain valuable structured data that is hidden in regular English sentences. This data is best exploited if available as a relational table that we could use for answering precise queries or for running data mining tasks. We explore a technique for extracting such tables from document collections that requires only a handful of training examples from users. These examples are used to generate extraction patterns that in turn result in new tuples being extracted from the document collection. In selecting documents for examination, users must be able to formulate reasonably precise queries. Queries that are too broad will greatly reduce the efficiency of information discovery efforts by overwhelming the users with peripheral information. In order to formulate efficient queries, a mechanism is needed to automatically alert users regarding potentially interesting information contained within the collection. The idea can better be explained into two sub topics. One is extraction of relations from natural plain text and another is frame discovery and formation of word-net from language text. In this paper we have tried to explain how to extract the different kind of relationship between the words with the help of a frame net analysis diagram of an annotation layer software.
Keywords- information discovery, extraction patterns, tuples, queries, frame discovery

Introduction

There is an increasing need for methods for efficiently identifying information of interest in large collections of text. For example:

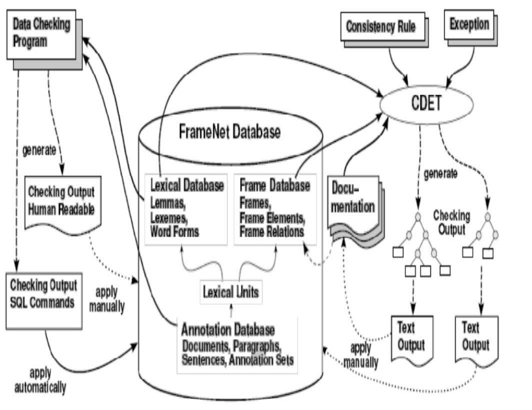
- Corporations wish to survey large collections of e-mails to identify illicit activities or those that may potentially generate liability.
- Legal firms need to review large collections of documents obtained during the discovery phase of litigation.
- Counterterrorism analysts have a requirement to rapidly review large amounts of incoming messages to identify information relevant to potential threats.

In general, humans must read at least some of the documents contained in collections in order to make definitive judgments about the presence or absence of new and interesting information. For large text collections, the key problem is to minimize the amount of information that must be read by a user in order to make such judgments. The key problem is that the user typically has only a very general idea of the specific contents of the collection. In order to efficiently select documents for analysis, the user must invoke reasonably precise queries. Text documents often hide valuable structured data. For example, a collection of newspaper articles might contain information on the location of the headquarters of a number of organizations. If we need to find the location of the headquarters of any organization, say, IBM, we could try and use traditional information-retrieval techniques for finding documents that contain the answer to our query [13]. Alternatively, we could answer such a query more precisely if we somehow had available a table listing all the organization-location pairs that are mentioned in our document collection. A tuple<o,l> in such table would indicate that the headquarters of organization o are in location l, and that this information was present in a document in our collection. Tuple <IBM, ARMONK>in our table would then provide the answer to our query. The web contains millions of pages whose text hides data that would be best exploited in structured form.

Frame Development and Frame Net Annotation

A lexical unit (LU) is a pairing of word with a meaning. Typically each sense of polysemous word belongs to a different semantic frame, a script like conceptual structure that describes a particular type of situation, object, or event along with its participants and props. For example the Apply_heat frame describes a common situation involving a COOK, some FOOD and a HEATING_INSTRUMENT, and is evoked by words such as bake, blanch, boil, broil, brown, simmer, stream etc. We call these roles frame elements (FEs) and the frame evoking words are LUs in the Apply_heat frame. Some frames are more abstract, such as change_position_on_a_scale, which is evoked by LUs such as decline, decrease, gain, plummet, rise, etc. and has FEs such as ITEM, ATTRIBUTE, INITIAL_VALUE and FINAL_VALUE.
In the simplest case, the frame-evoking LU is a verb and the FEs are its syntactic dependents:
(i) (Cook) Mathew fried (food) the fish (Heating_Instrument) in a heavy iron skillet.
(ii) (Item) ITC stock rose (Difference) $4 (Final_value) to $40.

However, event noun such as reduction in the cause_change_of_scalar_position frame also evoke frames:
….the reduction (item) of debt levels (value_2) to $500 million (value_1) from $2.5 billion
or objectives such as asleep in the Sleep frame:
(Sleeper) They were asleep (Duration) for hours.
The lexical entries for a predicting word, derived from such annotations, identifies the frame which underlies a given meaning and specifies the ways in which FEs are realized in structures headed by the word. Framenet annotations derive from two sources. In pursuing the goal of recording the range of semantic and syntactic combinatory possibilities of each word in which of it senses, we normally concentrate on a particular target LU and extract sentences from the different texts containing that LU. In another kind of work that represents a much smaller percentage of our overall annotations, we annotate running text. Full text annotation differs from sentence annotation mostly in that the sentences are chosen for us, so to speak, by the author of the text. The annotation of running text is also technically possible. Frame net lexicographers can one by one declare each word in a sentence of target, select a frame relative to which the new target is to be annotated, get a new set of annotation layers (frame element, grammatical function, phrase type) and appropriate frame element tags, and then annotate the relevant constituents. The core of the process has always been looking at attestations of a group of words that we believe to have some semantic overlap, and dividing these attestations into groups. Afterward we combine the small groups into large enough groupings to make reasonable frames at which point we may (equivalently) call the words targets, lexical units, or frame-evoking elements. In the past the criteria of such grouping have been informal and intuitive, but recently, the criteria have become more explicit. The basic semantic type for a frame element ought to be broadly constant across uses. If that is not so it suggests the need to posit distinct frame elements. In some cases, however, we still want to recognize a relationship between frame elements whose syntactic form suggests that they refer to ontologically different kinds of entities. For example, in I want [to win] compared with I want [an orange], both complements of the verb "want" have something to do with the desiring frame, but each of the complements directly refers to something rather different.
As a technical matter, the way in which FrameNet analyzes instances of a target predicate consists of marking up parallel aligned layers of annotation with appropriate label sets. The number of layers and the kind of information that can be recorded on them is technically unlimited. But in FrameNet's current practice the four core annotation layers are the Target, frame element (FE), grammatical function (GF), and phrase type (PT) layers. On the first, the(parts of the) target predicate are marked while on the latter three, labels are applied to the constituents expressing the frame elements of the target. The next-most important set of layers consists of the layers called other; a layer called either Noun, Verb, Adj, or Prep depending on the part of speech of the target (this layer is also often called the part-of-speech-specific layer); and the Sent(sentence) layer. A final group of layers includes, among others, layers holding labels related to part of speech (POS) and Named Entity Recognition (NER). Generally FrameNet is the collection of frames consisting different types of frame elements like Noun phrase (NP), Verb phrase (VP), Adverb phrase (ADP), Adjective phrase (AJP), Preposition phrase (PP) and FrameNet analyzes the instances of different layer of annotation like Frame Elements (FE), Grammatical Functions (GF) and Phrase Types (PT).


Data Checking Program
Consistency Rule
Exception
CDET
FrameNet Database
generate
Checking Output Human Readable
Lexical Database Lemmas, Lexemes, Word Forms
Frame Database Frames, Frame Elements, Frame Relations
Docu-mentation
generate
Checking Output
Checking Output SQL Commands
apply manually
Lexical Units
Annotation Database Documents, Paragraphs, Sentences, Annotation Sets
apply automatically
apply manually
Text Output
Text Output
Fig.1- Framenet Architecture

Related Works
In their work on semantic annotation, Mingcai Hong et.al. [1], addressed the issue of semantic annotation using horizontal and vertical contexts. OnTeA: (Semi-automatic Ontology based Text Annotation Method was developed by Michal Laclavık et.al. [2]. It describes a solution for the ontology-based text annotation tool to analyze a document or text using regular expression patterns and detects equivalent semantic elements according to the defined domain ontology. Fabio Ciravegna et al [3] developed a tool, Melita, for the definition and development of ontology-based annotation services which does beyond the dichotomy rule learning versus rule writing of classic annotation systems, as it allows adopting different strategies, from annotating examples in a corpus for training a learner to rule writing and even a mixture of them. In addition to this, there are a number of annotation tools and approaches such as CREAM [4, 5] or Magpie [1,

3] which provide users with useful visual tools for manual annotation, web page navigation, reading semantic tags and browsing [1, 5] or providing infrastructure and protocols for manual stamping of documents with semantic tags. The basic limitation of these systems is that they are geared towards information extraction and not knowledge discovery. As mentioned earlier, the result of an annotated text is used for further computer processing, for example, using semantic data in knowledge management [1, 3] or in Semantic Organization applications. This further processing for example can be captured in Text mining. Ronen Feldman, et.al. [6], developed a Document Explorer, a tool that implements text mining at the term level. Earlier works on mining association rules from text have explored the use of manually assigned keywords. Where, they used keywords as features for generation of association rules [7, 8]. The drawbacks of approaches that use manually assigned keywords are that: (1) it is time consuming to manually assign the keywords; (2) the keywords are fixed (i.e., they do not change over time or vary based on a particular user); (3) if the keywords are manually assigned, they are subject to discrepancy; (4) the textual resources are constrained to only those that have keywords. Several other researchers [9, 10, 11] applied existing data mining techniques to discover episode rules from texts, where Episode rule mining is used for language analysis because it preserves the sequential structure of terms in a text document. However, in our work we focus on the extraction of association rules that present the relations existing among the keywords in texts ignoring the order in which the keywords occur.

Experiment Methodology

We used the parser of Collins (1997)[12], a statistical parser trained on examples from the Penn Treebank, to generate parses of the same format for the sentences in our data. Phrase types were derived automatically from parse trees generated by the parser, as shown in Figure 2. Given the automatically generated parse tree, the constituent spanning
the same set of words as each annotated frame element was found, and the constituent's
nonterminal label was taken as the phrase type. In cases where more than one constituent matches due to a unary production in the parse tree, the higher constituent was chosen.

The matching was performed by calculating the starting and ending word positions for each constituent in the parse tree, as well as for each annotated frame element, and matching each frame element with the parse constituent with the same beginning and ending points. Punctuation was ignored in this computation. Due to parsing errors, or, less frequently, mismatches between the parse tree formalism and the FrameNet annotation standards, there was sometimes no parse constituent matching an annotated frame element. This occurred for 13% of the frame elements in the training set. The one case of systematic mismatch between the parse tree formalism and the FrameNet annotation standards is the FrameNet convention of including both a relative pronoun and its antecedent in frame elements. Mismatch caused by the treatment of relative pronouns accounts for 1% of the frame elements in the training set. During testing, the largest constituent beginning at the frame element's left boundary and lying entirely within the element was used to calculate the features. We did not use this technique on the training set, as we expected that it would add noise to the data, but instead discarded examples with no matching parse constituent. Our technique for finding a near match handles common parse errors such as a prepositional phrase being incorrectly attached to a noun phrase at the right-hand edge, and it guarantees that some syntactic category will be returned: the part-of-speech tag of the frame element's first word in the limiting case.

Algorithm used for extracting new tupples using a set of patterns

```
GenerateTuples(Patterns)
For each text segment
(1) {<o,l>,< ls,t1,ms,t,rs >}=

        CreateOccurrence(text_segment);
        TC = <o,l>;
        SimBest = 0;
        For each p in Patterns
(2) sim = Match(<ls,t1,ms,t2,rs >,p);
        if (sim>=Tsim)
(3) UpdatePatternSelectivity(p,TC);
        if(sim>=SimBest)
                SimBest=sim;
                PBest=p;
        if(SimBest>=Tsim)

        CandidateTuples[TC].Patterns[PBest] =
SimBest;
return CandidateTuples;
```

Results

Results on identifying frame elements (FEs), including partial matches. A total of 7,681 constituents were identified as FEs, and 8167 FEs were present in hand annotations, of which matching parse constituents were present for 7,053(86%). When the automatically identified constituents were fed through the role labeling system described above, 79.6% of the constituents that had been correctly identified in the first stage were assigned the correct role in the second, roughly equivalent to the performance when assigning roles to constituents identified by hand.

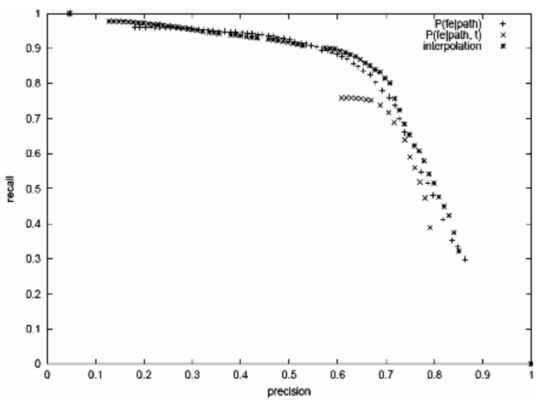


Fig. 2- Precision/recall is plotted for various methods of identifying frame elements.

Conclusion

The results of this experiment are very encouraging. They demonstrate that the technique employed is capable of identifying numerous associations of interest in large text collections in a completely automated fashion. The approach could be used to provide a rapid overview of large volumes of text. In a dynamic collection environment it could be used to automatically alert users when text has been acquired that contains information of interest. The approach described here has a number of important characteristics that make it particularly well-suited for intelligence analysis applications:

(1) It is independent of topic, genre, or language.

(2) Identified relationships represent the aggregate implications of high-order associations. For this reason, the approach is capable of identifying quite subtle relationships.

(3)The user has complete flexibility in specifying items of interest. Any point in the LSI representation space can be used as a basis for defining a row or column in the comparison matrices. In particular, a textual description of an entity or concept of interest may be chosen as a definition of an item of interest. That description does not have to have been derived from the documents indexed. There need only be some minimal degree of overlap in terminology between the description and the aggregate terminology employed in the documents used to generate the representation space.

(4)The technique has significant utility in identifying possible use of aliases.

(5)There is no need for predefined auxiliary structures such as taxonomies or ontologies. Nor is there need for any linguistic analysis, other than that contained in the entity extraction software

References

[1] Mingcai Hong, Jie Tang, and Juanzi Li: Semantic Annotation using Horizontal and VerticalContexts, 2004.

[2] Michal Laclavik1, Martin Seleng1, Emil Gatial1, Zoltan Balogh1, Ladislav Hluchy1: Ontology based Text Annotation, 2004

[3] Fabio Ciravegna, Alexiei Dingli, Jose‘ Iria, and Yorick Wilks: Multi-strategy Definition of Annotation Services in Melita, 2002

[4] Domingue J., Dzbor M.: Magpie: supporting browsing and navigation on the semantic web. IUI '04, pages 191-197, New York, NY, USA, 2004. ACM Press. ISBN 1-58113-815-6.

[5] Handschuh S., Staab S.: Authoring and annotation of web pages in cream. In WWW '02, pages 462-473, NY, USA, 2002. ACM Press. ISBN 1-58113-449-5. doi:

[6] http://doi.acm.org/10.1145/511446.511506.

[7] Uren V. et al.: Browsing for information by highlighting automatically generated annotations: a user study and evaluation. In K-CAP '05, pages 75-82, NY, USA, 2005b. ACM Press. ISBN 1-59593-163-5

[8] Ronen Feldman, Moshe Fresko, Haym Hirsh, Yonatan Aumann,* Orly Liphstat, Yonatan Schler, Martin Rajman: Knowledge Management: A Text Mining Approach: Proc. of the 2nd Int. Conf. on Practical Aspects of Knowledge Management (PAKM98) Basel, Switzerland, 29- 30 Oct. 1998, (U. Reimer, ed.)

[9] Feldman R. and Dagan I., “Knowledge discovery in textual databases (KDT)”, in Proc. 1st Int. Conf. on Knowledge Discovery and Data Mining, 1995.

[10] Feldman R. and Hirsh H., “Mining associations in text in the presence of background knowledge,” in Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining, Portland, USA, 1996.

[11] Ahonen H., Heinonen O., klemettinen M., and Inkeri V., “Mining in the phrasal frontier,” in Proc. PKDD’97.1st European Symposium on Principle of data Mining and Knowledge Discovery, Norway, June, Trondheim, 1997.

[12] Ahonen H., Heinonen O., Klemettinen M., and Inkeri V., “Applying data mining technique for descriptive phrase extraction in digital document collections” in Proc. of IEEE Forum on Research and technology Advances in Digital Libraries, Santa Barbra CA, 1998.

[13] Collins, Michael. 1997. Three generative, lexicalised models for statistical parsing. In Proceedings of the 35th Annual Meeting of the ACL, pages 16–23, Madrid, Spain.

Table 1- Matching of overlaps

Type of Overlap	**Identified Constituents**	**Number**
Exactly matching boundaries	63%	5326
Identified constituent entirely within true frame element	8	653
True frame element entirely within identified constituent	7	593
Both partially within the other	0	23
No overlap with any true frame element	13	997