Extracting semantic sets from the World Wide Web using unsupervised learning

Chinnappa Guggilla, Sadiq Mohamed, Sridhar Vardarajan

Applied Research Group, Satyam Computer Services Ltd

Indian Institute of Science Campus, Bangalore - 560012, India

Abstract

The exponential growth of World Wide Web poses many challenges for web researchers in the process of extracting qualitative knowledge. Many modern knowledge acquisition approaches exploit the redundancy of the Web to extract semantic information. In this paper, we propose a statistics based method of finding a set of semantically related terms, given a small set of seed items. We used an Expectation-Maximization algorithm for unsupervised learning to extract semantic clusters by making use of redundant information available in the web pages. We evaluated and compared our results with similar applications, Google sets and SEAL (set expander for any language). We also present a method for labeling the sets or clusters extracted from the Web given the seed terms. Labels are useful for building the concept taxonomies.

Keywords: World Wide Web, Expectation-Maximization, unsupervised learning, semantic clusters, taxonomies.

1. Introduction

Due to the ease of creating bundles of web pages, the World Wide Web has been growing exponentially, containing heterogeneous data. Over the years, the web has conglomerated with several billions of pages. In recent years, many search engines like Google, Yahoo, AltaVista, etc have emerged, and struggling to provide the quality results to end users. In the last two years, several semantic search engines such as PowerSet, 2006; Hakia, 2004; Lexxe, 2006; Cognition Search, 2006; have also emerged to deliver the semantically related information for the user query. These are making use of Natural Language Processing (NLP) and Semantic Web technologies.

To produce the semantically relevant results to the end user, there is a need for using ontology and semantic clusters or clusters of concepts. We need to have large scale, established and standardized ontologies in order to make sense of the information. But such a categorization is already encoded in the structure of the web. In fact, the added benefit from such resource-intensive ontology development and characterization is not readily apparent. One important characteristic of the web is that information is replicated and scattered across the web and is available from independent resources. We will have more confidence in extracted information if the same information is present on different websites.

Our approach is the way to browse the web's implicit ontology. And it's also a great way to mine the web. What we do is simple: Just enter some terms which you already think of as instances of some class. Our system then returns you the other instances of that class along with the class (cluster) name. Mainly our approach utilizes the structure of the web pages

along with replicated information to generate the semantic clusters which are the building blocks for ontology. Tables, lists, bulleted lists are explicitly represented in the html page, along with the comma separated text and 'and' separated phrases. These are the text blocks which will be extracted from all the web pages. We prepare web corpus from these text blocks and it will be used as training corpus for our algorithm. We use the Expectation-Maximization (EM) unsupervised algorithm for learning the semantic cluster probabilities from the web corpus. These cluster probabilities are estimated from the table, list, for example text blocks and comma separated text blocks using Bayesian classification algorithm.

Given a seed set of terms, semantic clusters (sets) are calculated from the seed co-occurrence probabilities. These probabilities are estimated from the web corpus. Mainly in this paper, we try to solve two problems using the World Wide Web.

1. To predict more items in the same category, given a small set of seed items.
2. To label the semantic clusters extracted, given the set of seed examples.

This paper is organized as follows. In section 2, we start with describing the related work. In section 3, we introduce the basic ideas of our approach with an example. Section 4, describes the basic learning algorithm for extracting the clusters. In section 5 we discuss the method for labeling the set. In section 6 we evaluate our work and compare the results with Google Sets and SEAL, (Set Expander for Any Language) 2005. In section 7 we conclude our work with the possible future directions.

2. Related Work

The functionality of the Google sets, 2004; is the closest to our system. The Google system takes few example seeds and finds more instances that are related to the seed set. To the best of our knowledge, there is no literature describing the implementation of Google sets. We evaluated our work by comparing the results returned by Google sets and SEAL.

Dilip Rao and Deepak Khemani, 2007; proposed method is also similar to the Google sets. The idea is: given a set of few examples their system tries to find more instances that are similar to the seed set using bootstrapping. Their system uses only comma separated text blocks for finding terms related to the seed terms. This system also proposed a method to form the hierarchy of terms. In comparison with this system, our system uses not only comma (,) separated lists but also tables, lists and for example text blocks to generate related terms for the given seed set. Since tables and lists store the semantically related information, these elements serve better purpose in generating relevant terms for the seed terms. Unlike Google and this system, our system labels the semantic clusters or sets. This labeling is important in generating the semantic networks and ontologies.

SEAL, 2005; is a system, which takes at least two seed items, and predicts the relevant set. If we give seed set in one language, it generates the relevant seed set corresponding to that language. It works across languages. It performs well if seed set contains only named entities.

Marti Hearst, 1992; applied regular expression patterns, called lexico-syntactic patterns, for extraction of semantic relations from the text. His underlying idea is very simple and can be easily applied to the web to derive relational information from the web. Normal regular expressions capture recurring expressions and map the results of the matching expression to a semantic structure, such as taxonomic relations between concepts. Consider the following example:

Fruits like Apple, Banana and grapes

Above text block allows us to generalize that Apple, Banana, Grapes and Pine apple are instances of fruits.

Serge Brin, 1998; extracts instances of books and authors (patterns and relations) from the World Wide Web. He used patterns to extract relations and from relations he extracted patterns. Geleijnse and Korst, 2005; used specialized patterns to derive actors and director of movie patterns.

Eugene Agichtein and Luis Gravano. 2000; system tries to learn general extract patterns starting from few examples. They extracted the context surrounding the example pattern to extract context templates for extracting more tuples. The extracted tuples are used again to discover the more templates and the process continues iteratively till sufficient instances are retrieved.

Andrew Rabinovich, Andrea Vedaldi, et al., 2007; automatically labels the objects in an image. They used the co-occurrence of object labels in the training set, to accurately label the object in an image. They also used Google sets for generic context information to disambiguate the labels.

3. Example

In this section, we briefly describe our idea of generating semantically related terms given a seed terms from the web pages through examples. Assume that we are searching for countries like India, Pakistan and Srilanka in the Google search engine and it returned the results as shown in Figure 1. To generate more countries like India, and Pakistan we will use the comma separated lists from the WebPages as shown in figure 1. We can also extract more related terms for the given seed terms using table, list and for example text blocks in which terms are semantically related. The group of semantically related terms is called a set or a cluster. These sets are labeled using category name or class name and these labels can be used to build taxonomies or to generate domain ontology.

Following are some of the examples which will better depict our basic idea.

Most of the current image labeling systems are not categorizing the objects in an image accurately due to the lack of semantic context. For example if an image contains, tennis bat, tennis ball (which is in yellow color), ground, and a tennis player objects in it, state of the art image labeling systems label the tennis ball incorrectly as ‘lemon’ based on the color. But if we provide external semantic context in which tennis bat, player, ground and ball are co-occurred, most likely that the yellow ball will be labeled as ‘ball’ instead of lemon. It means that the tennis ball object is semantically related with player, ground and bat.

If we want to see, who all are competitors of our company, just input name of the company we work for along with one competitor? We will get list of competitors.

If we want to see the movies we haven’t seen or other of our favorite movies just enter some of our favorite movies. We can get suggestions for books by entering some of our favorite authors or titles

We can build an online quiz by just inputting one correct choice, and letting our system to predict other wrong choices.

Amazon. com: Birds of India, Pakistan, Nepal, Bangladesh, Bhutan...
Birds of India, Pakistan, Nepal, Bangladesh, Bhutan, Sri Lanka and the... Birds of India, Pakistan, Nepal, China, Bhutan, Sri Lanka and the Maldives...
www.amazon.com/Birds-Pakistan-Bangladesh-Bhutan-Maldives/dp/0691049106- 180k-Cached - Similar pages

Fig. 1. An example of search result for the countries like India, Pakistan and Srilanka

4. Term inducing using unsupervised learning algorithm

Our basic problem is that given a seed set of examples, predicting other items in the same category as that of examples provided. This means that the system creates a set of word associations from the starter example. The predicted terms are called a set or cluster of the terms. For example, if we are given green, blue and purple as the seed terms, we would like to predict related terms such as red, white, black and yellow belonging to the same category of seed set. Initially, we consider only seed terms, there after growing the set without knowing the category or set name (domain name) of seed terms. Though we can use domain specific method for finding domain related words, we are proposing a domain independent method to find any kind of semantically related terms for the given seed set.

We know that generally, most of the web pages contain textual data in the form of comma separated blocks, in table structured blocks and list blocks. Naturally, if terms in a sentence are separated by commas (,), we can say that those terms belong to the same category or semantic cluster. Similarly if the terms are stored in a table, we can say that those terms are more semantically related. Likewise, we can say that if the terms are listed in the list tag, these terms are also related. Similarly, we also consider 'for example' and 'and' separated text blocks to find the semantically related terms for the given seed terms.

Our proposed idea is simple: If the seed term(s) has co-occurred with the term in the above specified text blocks, it is more likely that the seed term and the co-occurred term are related. This is called term co-occurrence. If a seed term(s) co-occurs with any other term in the text blocks more number of times, it is more likely that the two terms are related and the other term becomes the new seed term.

It is assumed that the web corpus is prepared from the top 100 web pages, which are retrieved from the Google search engine for the given seed term. The web corpus contains only comma separated text blocks, table blocks, list blocks and "for example" blocks. Now, given a seed term, How do we calculate term co-occurrence probabilities to select the new seed from the web corpus?

Let us denote the seed term as 'st' and the new term as 'nst'. To induce (predict) a new seed, from the given seed terms, we need to have a model **P(nst | st)** which estimates the conditional probability of any new seed, given the seed term. Given a seed term 'st', we predict the new seed term 'nst' that maximizes P(nst|st), i.e. the "most likely" seed term, out of all such seed terms, which yields the highest value for P(nst|st).

Using Bayes' rule, we can write the above problem as follows:

$$p(nst \mid st) = \frac{p(nst) \times p(st \mid nst)}{p(st)} \quad (1)$$

We can rewrite the expression for the most likely seed term as following:

$$p(nst \mid st) = \arg\max_{nst} p(nst) * p(st \mid nst) \quad (2)$$

That means the most likely seed term 'nst' maximizes the product of two values. Here, P(st) is same for all seed terms.

We will focus only on computing **p(nst | st),** i.e., the term co-occurrence modeling. The term co-occurrences are estimated from the web corpus.

The co-occurrence model uses the simple idea of co-occurrence of **st** and **nst** in the web corpus: if **st** and **nst** tend to co-occur in the text blocks of web corpus, they are likely to be semantically related to each another. We can also say that **st** and **nst** are likely to be distributional similar terms.

4.1. Parameters for learning term co-occurrence

Term co-occurrence parameter, which is denoted by P (nst | st) is calculated from the following sub parameters. "For example" text blocks are merged with the comma separated text blocks to reduce the number of parameters for learning.

4.1.1. Comma-parameter

Given a seed term of a set, the new seed is induced from the comma separated terms using seed co-occurrence probability measure p(nst|st). This parameter is denoted by C(nst | st).

$$p(nst \mid st) = \coprod_{i=1}^{l} C(nst_i \mid st) \tag{3}$$

Where 'l' denotes the number of terms in the comma separated text blocks. And 'nsti' denotes 'i'th term in the comma text.

4.1.2. Table-parameter

Given a seed term of a set, the new seed is also induced from the table data using seed co-occurrence probability values. This parameter is denoted by T(nst | st).

$$p(nst \mid st) = \coprod_{j=1}^{m} T(nst_j \mid st) \tag{4}$$

Where 'm' denotes the number of terms in the table text blocks and 'nstj' denotes 'j'th term in the table text.

4.1.3. List-parameter

For the given seed term, the new seed is discovered from the list terms using seed co-occurrence. This parameter is denoted by L(nst | st).

$$p(nst \mid st) = \coprod_{k=1}^{n} L(nst_k \mid st) \tag{5}$$

Where 'k' denotes the number of terms in the list text blocks and 'nstk' denotes 'k'th term in the list text.

Now we can combine (1), (2), (3) to calculate the final term co-occurrence parameter as following.

$$p(nst \mid st) = \prod_{i=1}^{l} C(nst_i \mid st) \times \prod_{j=1}^{m} T(nst_j \mid st) \times \prod_{k=1}^{n} L(nst_k \mid st) \tag{6}$$

New seed will be discovered from the max value of term co-occurrence parameter p(nst|st) using the following equation.

$$p(nst \mid st) = \arg\max_{nst} \prod_{i=1}^{l} C(nst_i \mid st) \times \prod_{j=1}^{m} T(nst_j \mid st) \times \prod_{k=1}^{n} L(nst_k \mid st) \tag{7}$$

How do we calculate comma C(nst|st), table T(nst|st) and list L(nst|st) parameters from the web corpus? How do we discover new seed terms from these parameters? The parameters will be learnt from web corpus using Expectation-Maximization algorithm.

4.2. E-M Algorithm

For calculating the parameters mentioned above (comma, table and list), we use a generative algorithm called Expectation Maximization (EM) for training. The EM algorithm guarantees an increase in likelihood of the model in each iteration, i.e. it is guaranteed to converge to a maximum likelihood estimate.

Web corpus is used as the training data. Let us assume that the seed set is 'S'. The term co-occurrence parameter P(nst | st) is learnt during training using expected parameter counts. After training, a new term is induced from the term co-occurrence parameter. Let the number of iterations during training be 'N', where 'N' indicates the limit for the number of terms included in the set. The iterative EM algorithm corresponding to the term co-occurrence problem can be described as:

Step-1: Input the seed set 'S' containing at least one term**.**

Step-2: Select one seed term from 'S', say 'st' and query the search engine and prepare the web corpus from comma, table and list text blocks matching the query term in 'st'

Step-3: Collect all word types from the web corpus corresponding to the comma, table and list text blocks. Let us say we found 'l', 'm', 'n' terms in corresponding text blocks. Collect all new words '**nst'** those co-occur at least once with 'st'.

Step-4: Initialize the comma, table and list parameters uniformly (uniform probability distribution), i.e., any target word probably can be the semantically related to seed term.

C(nst|st) = T(nst|st) = L(nst|st) = 1/total number of co-occurring terms in all text blocks

Step-5: Update the expected counts of comma, table and list parameters from the corresponding probability values for the entire web corpus.

For each term in comma block, table block, and list block calculate the corresponding totals

totalc+ = C(nsti | st), totalt + = T(nstj | st), totall + = L(nstk |st)

And update expected counts of corresponding parameters as shown in below

C(nsti |st) = C(nsti | st) /totalc, T(nstj |st) = T(nstj | st) /totalt,

L(nstk |st) = C(nstk| st) /totall

Step-6: Re estimate the corresponding parameter values and calculate the co-occurrence seed term parameter using following equation.

$$p(nst \mid st) = \prod_{i=1}^{l} C(nst_i \mid st) \times \prod_{j=1}^{m} T(nst_j \mid st) \times \prod_{k=1}^{n} L(nst_k \mid st) \quad (8)$$

Step-7: select the new seed term from the following equation and update the seed set.

$$p(\underset{nst}{nst} \mid st) = \arg\underset{nst}{max} \prod_{i=1}^{l} C(nst_i \mid st) \times \prod_{j=1}^{m} T(nst_j \mid st) \times \prod_{k=1}^{n} L(nst_k \mid st) \quad (9)$$

Add the new seed term 'nst' to 'S'.

Step-8: Repeat steps 2-8 until the threshold 'N' meets.

5. Assigning label (category name) to the generated set

After generating the semantically related set (cluster) from the seed set, the set will be labeled by the category to which it belongs to. For example if the set containing India, Pakistan, USA and UK is generated, then it has to be labeled with the category 'country' name.

5.1. Labeling using Word Net

Christiane Fellbaum, 1998, Word Net; is used to get the glosses of terms. Randomly after selecting some terms from the seed set, glosses are extracted. Stop words are eliminated from the glosses. Then we calculate the overlapping terms which are commonly occurred from all the glosses. The common term is named as label or category of the new set. If more than one common term (label) is generated, only one label among the common terms is selected based on hypernyms of random seed terms. The labels are matched with the hypernyms of seed terms, if any label is matched with more number of seed terms' hypernyms, that label is selected as final label or category of the new set (cluster).

5.2. Labeling by the EM-algorithm

The generated set can also be labeled or given a category name in the final iteration of the EM-algorithm. In the final iteration, some random seed terms are selected from the grown seed set and matched with the corresponding text blocks' terms in the web corpus. We are calculating the seed - new term co-occurrence probability for all randomly selected seed terms except stop words. The term which has co-occurred with all the seed terms and having the maximum co-occurrence probability is named as label or category of the new set or cluster.

6. Evaluation

6.1. Evaluation of Term inducing

The results generated by unsupervised learning algorithm are evaluated using well known IR measures: Precision, Recall and F-measure. Our system results are compared with Google sets and SEAL. Both systems allow at least 2 seed terms, and returns smaller sets. For comparing with our system results, we use the 'smaller set' option in Google sets and SEAL to check with only top most related terms. According to the 'COMMA' system, Precision and Recall are defined as follows for evaluation.

Let P, G and S be the result set returned by our System, Google and SEAL Respectively, correct(P) is defined as the subset of P that contains only semantically related to the seed terms. Similarly correct (G) and correct(S) are defined. Since we don't have access to exhaustive set of terms related to the seed set, Precision and Recall are defined with respect to the union of the correct sets retrieved by any two systems.

Precision (P) = |correct (P)| / |P|

Recall (P) = |correct (P)| / |correct (P) Ụ correct (G)|

Precision (G) = |correct (G)| / |G|

Recall (G) = |correct (G)| / |correct (G) Ụ correct (P)|

Precision(S) = |correct (S)| / |S|

Recall(S) = |correct(S)|/|correct(S) Ụ correct (P)|

Similarly F-measure is defined as: 2*P*R/ (P+R)

The number of seed terms presented to any two systems is limited to two. For each category, the number of terms discovered by our system, Google and SEAL (represented as Sy, Go, and SE in tables) and the number of overlapping terms are listed in table1. Precision, Recall and F-measure results corresponding to table1 are listed in table 2.

6.2. Evaluation of label generation

The labels assigned to the four sets are generated from the four seed sets. We manually checked the labels assigned by our system. For four categories of sets, the assigned labels for corresponding sets are listed in table 3. One observation in labeling of sets is that Word Net will not supply glosses for new terminology present in the web pages. In this case we will use other terms in the set to get glosses, from which we are able to get overlapping terms.

In some cases, two labels are assigned to the same set (as shown in Table 3) as more than one overlapping term is generated. We also considered the label generated by EM- algorithm to label the Set, if label is not assigned by Word Net approach.

7. Conclusion and Future Work

In this paper, we presented a novel approach for generating semantic sets and labeling the set, from the World Wide Web, based on the observation that web page contain comma separated text, table data and list data. In order to identify more semantically related terms, we used unsupervised learning algorithm on web corpus. We evaluated our results for four seed categories, and compared with Google sets and SEAL implementation. The high precision of our method is attributed to relying on additional table and list data, along with the comma separated text. Our method performs better than the Google sets and SEAL implementation. We used Word Net glosses to generate labels for the semantic sets. Future work could include identifying more text blocks in a web page, from which semantically related terms can be generated, building the semantic networks and ontology from semantically related sets or clusters, and given some phrases, as starter examples, generating the semantically related phrases.

Category	No. of terms			Overlap terms	
	Sy	Go	SE	Go	SE
Color	11	11	8	9	6
Fruit	14	15	16	5	6
Company	13	15	31	6	7
Sports	10	12	10	9	10

Table 1. Results of term inducing: number of terms in a set and overlapping terms.

Precision			Recall			F-measure	
Sy	Go	SE	Sy	Go	SE	Sy	Go
100	100	87.5	50	50	38.8	**66.6**	66.6
50	50	56.2	50	50	56.2	**50**	50
61.5	50	29	53.3	46.6	52.9	**57.1**	48.2
100	100	100	45.4	54.5	50	**62.4**	70.5

Table 2. Precision, Recall, F-measures of three systems.

Seed terms	Label assigned
Green, blue	Color
Apple, banana	Fruit/food
Google, amazon	Organization/company
Cricket, football	Game/sports

Table 3. Labels assigned by our system for the expanded Set.

References

Agichtein E. and Gravano L. (2000). Snowball: Extracting relations from large plain-text collections. In *Proceedings of ACM DL.*

Brin S. (1998). Extracting patterns and relation from the world wide web. In *Proceedings of the WebDB workshop at 6th International Conference on Extending Database Technology.*

CognitionSearch. (2006). A meaning based search engine. http://cognitionsearch.com/

Fellbaum C. (1998). *WordNet: An electronic Lexical Database*. MIT Press.

Geleijnse G. and Korst J. (2005). Automatic ontology population by googling. In *Proceedings of the Seventeenth Belgium-Netherlands Conference on Artificial Intelligence*, pages 120-126.

Google. Google sets, (2004) http://labs.google.com/sets.

Hakia. (2004) A semantic search engine. http://www.hakia.com/.

Hearst M. A. (1992). Automatic acquisition of hyponyms from large text corpora. In *Proceedings of COLING*, pages 539-545.

Lexxe. (2006). A semantic search engine http://www.lexxe.com/.

Manning C. D., Scheutze H. (1990). A book on Foundations of Statistical Natural Language *Processing Language Arts & Disciplines*, 680 pages.

Powerset. (2006). A semantic Search Engine. http://www.powerset.com/

Rabinovich A., Vedaldi A., Galleguillos C., Wiewiora E., Belongie S. (2007). Objects in Context. *The 11th IEEE International Conference on Computer Vision in Rio de Janeiro, Brazil, CCV*.

Rao D. and Khemani D. (2007). Discovering semantic similarity from the World Wide Web. *Workshop on Text-Mining and Link-Analysis*, IJCAI.

Rosario B. and Hearst M. (2001). Classifying the Semantic Relations in Noun Compounds via a Domain-Specific Lexical Hierarchy. In the *Proceedings of EMNLP '01*, Pittsburgh, PA.

Set Expander for Any Language (SEAL). (2005). http://rcwang.com/seal/