On Automated Hypernym Hierarchy Construction Using an Internet Search Engine*

Sergey Afonin

Institute of Mechanics, Moscow State University,
119192, Michurinskij av., 1, Moscow, Russian Federation
serg@msu.ru

Abstract. In this paper we propose an approach for automatic construction of concept hierarchies from the snippets returned by Internet search engines using a number of well known techniques. We use surface lexical patterns to construct a set of candidate hypernyms of a given term and additional filtering that is based on both lexical patterns and distributional analysis. Preliminary experimental results for real life English examples are presented.

Keywords: ontology learning, concept hierarchy, hypernyms, lexical patterns.

1 Introduction

Ontology is an ambiguous notion. In philosophy it goes back to Ancient Greeks and refers to the study of nature, or more generally, to the study of what might exist (Smith, 2004). In the context of computer science, terminological, information and knowledge anthologies (van Heijst *et al.*, 1997) were introduced. Later two ontologies deal with *concepts*, while terminological ontologies are focused on natural language *terms* that may or may not be directly mapped into concepts. In contrast to information ontologies, knowledge ontologies use some formal languages to represent sematic relations between concepts.

Terminological ontologies, such as WordNet, contain information about hyponym-hypernym (subtype-supertype, or IS-A relation), synonymy, and other semantic relations between terms. Such information can be used in variety of natural language processing applications, including query reformulation (Jones *et al.*, 2006; Chirita *et al.*, 2007), text summarization (Dang *et al.*, 2008), categorization (Li *et al.*, 2009), query answering (Lopez *et al.*, 2007), and many others. Unfortunately, good quality ontologies are mostly manually created and they exist for a limited number of domains. Moreover, it is difficult to support an ontology for rapidly changing domains when new concepts or semantic relations frequently appear, so ontology construction and updating become a bottleneck for many applications. A possible solution to the above problem is an automatic or semi-automatic *ontology learning*.

There is a lot of work has been done in the area of ontology learning during past two decades. While there exist many types of semantic relations between concepts and many different sources of information (e.g. databases, structured text, dictionaries) the problem of concept hierarchy learning from unstructured texts may be considered as the main topic in the area. An importance of this problem can justified by the fact that concept hierarchy forms a core of every ontology and that unstructured text is the most frequent data format.

Automated ontology learning from text is a challenging task. It is well known that the set of concepts and terminology varies between different languages and cultures. For example, in many northen languages there exist a lot of terms describing specific conditions of snow, that

* This work was supported in part by the Russian Foundation of Basic Research under grant 09-07-00366a.

can not be easily translated into other languages. Even in the same language some terms can be treated differently. For example, in the UK the word *city* means a settlement of high importance, differentiated from town or village by size, population density, or status. In the US almost all settlements are cities. One can find hundreds of Internet pages containing statements like "Canyon city is a small village".

Most of the work on concept hierarchy learning can be divided into two classes. The first class consists of work that are based on Harris' *distributional hypothesis* (Harris, 1968) which states that semantically similar terms tend to occur in similar contexts. For instance, one can expect that contexts of words *car* and *automobile* will contain many common words. Although the semantic similarity can be measured by means of statistics it is hard to describe the reason of high similarity. An example of statistical approach application to ontology learning is (Sanderson and Croft, 1999) which based on subsumption hypothesis, stating that the set of documents where two terms co-occur is a subset of the set documents where their hypernyms co-occur.

The second class of work, which the current work belongs to, uses *lexical patterns* (Hearst, 1992) that reflect specific semantic relations between terms. In particular, in order to estimate IS-A relation between terms A and B (e.g. hyponym-hypernym) one can search a corpus for *"B is a A"*, *"B is a kind of A"*, or *"B, C and other As"*. If such expressions exist in a corpus then one can assume that term A is a hypernym of B and C. Similar patterns were found for PART-OF relation (Berland and Charniak, 1999; Girju *et al.*, 2003). This approach is based on the assumption that in a sufficiently large corpus one can find a "definition" of term B written in one of above forms. The set of semantic pattern can be either manually coded, or automatically constructed by means of some machine learning technique. In contrast to statistical approaches, lexical patterns may deduce a hypernym from single occurrence. It is worth noticing that lexical patterns approach outperforms many statistical methods for semantic similarity measurement as well (Bollegala *et al.*, 2009). This is a surprising result because it is not evident what lexical patterns can reflect similarity between terms like *nail* and *hammer*, or *group* and *homomorphism*.

Lexical pattern approach suffers from two problems, namely, *sparsity of patterns* in real life texts (which is also a strong point), and *noisy output*. There are hundred thousand pages containing the pattern "robbery is a", but only a hundred of pages containing the pattern "oroshi is a". In the later case there exist a page with the correct definition, but it is given in a different form: "Oroshi is a word with a very specific meaning: wind blowing down from the mountain". Such snippet will produce an erroneous hypernym *oroshi - word*, that can be considered as a noise. For more rare terms no required patterns could be found at all. A combined "pattern-statistical" approach was very recently proposed in (Drumond and Girardi, 2010).

Our goal is development of a method for automatic hypernym learning from natural texts. Using lexical patterns approach and an Internet search engine as a mean for large corpus access we compute a set of candidate hypernyms of a given seeding term. For each candidate hypernym we compute the set of possible hypernyms (that are in some sense are neighbors of the seeding term) using the so-called double anchoring hyponyms discovery (Kozareva *et al.*, 2008). Finally we apply a filter that removes noisy seeding term's hypernyms. The filtering technique is based on the assumption that *the set of hyponyms of a correct hypernym of the seeding term should contain tarms that are sematically similar to the seeding term*. This is performed by means of statistical semantic similarity measurement and clustering. It is worth noticing that we do not require that all hyponyms should be semantically similar, so the noisy outut that can be generated by both double anchoring hyponym discovery and statistic similarity measure is not a problem.

The structure of the paper is the following. In the next section we briefly recall some useful results on applications of lexical patterns to ontology learning and semantic similarity measurement. In Section 3 the proposed approach is described. Section 4 contains evaluation results and in the last section we discuss the results and possible directions of future work.

2 Lexical Patterns and Related Work

Lexical patterns were introduced in (Hearst, 1992) as a mean for automatic IS-A (or hyponym-hypernim) relation mining. It was found that if two terms A and B are in this relation then they occur in a context like *A is a (kind of) B*, so a search for such pattern may yield a list of possible hyponyms. In contrast to statistical methods, lexical patterns approach allows to discover hyponym (or hypernym) of a given term from single occurrence in a corpus.

There is plenty of work based on the idea of lexical patterns, but we mention here the *double-anchoring* approach to hyponym discovery only, introduced in (Kozareva *et al.*, 2008). If one needs to construct a set of hyponyms of a given term, say H, provided that one hyponym, say A, is known, then it is possible to search a corpus for expression like *Hs such as A and* *. Doubly-anchored patterns serve two purposes: putting A into the query along with H (1) increases the likelihood of selected terms, and (2) eliminates possible ambiguity of H and/or A.

An attractive property of the above methods is that they could be implemented using commercial Internet search engines that makes it possible to run the algorithms on huge corpus. At the same time, capabilities of search engines are limited by their query languages that are not powerful enough to solve complex natural language processing tasks. In case when whole documents are available for processing additional information could be used. For example, in (Shinzato and Torisawa, 2004) the structure if HTML documents was utilized. The key assumption states that terms appearing in an HTML enumeration list could be hyponyms of some term, and this hypernym term is likely to appear in the text preceding to the list. Large text corpora in connection with lexical patterns used for solving other problems, like automatic construction of attribute words (Tokunaga and Torisawa, 2005).

As it was already mentioned, the proposed method for hypernym discovery assume *semantic similarity* between hyponyms of a term. Quantitative semantic similarity estimation is a well known and challenging problem. A number of "simply looking" methods that estimate similarity by means of statistical distribution of terms analysis were proposed, e.g. the Normalized Google Distance, Pointwise Mutual Information and their context-aware versions (Cilibrasi and Vitanyi, 2007; Gligorov *et al.*, 2007), but it seems that all such methods fail on ambiguous words. For ambiguous words clustering approach, such as one presented in (Pantel and Lin, 2002), produce consistent results, but clustering is computationally expensive. Taking into account specific properties of the problem we have to solve (we should compute semantic similarity between terms that assumed to be hyponyms of some common term), we narrowed our attention to the already mentioned hyponym-specific solutions.

3 Hypernym Construction Algorithm

Hypernym construction consists of three major steps, common to many natural language processing algorithms. They are:

- hypernym candidates set construction;
- candidates filtering ("termness" estimation);
- hypernym validity estimation.

In this section we describe how these steps are implemented. For Internet search tasks we use the YahooBOSS API[1].

3.1 Hypernym Candidates Set Construction

For initial hypernym candidate set construction corresponding to the seeding term *term* we use query of the form *"term is a *"*, where the whole expression is enclosed into quotation marks

[1] http://developer.yahoo.com/search/boss

forcing the engine to search for exact matching[2]. The actual query differs from this one in the following.

Let us consider the query *"Paris is a *"* expected to return such definitions of Paris as *city* or *capital*. In real search engine output we find a lot of the form *Hotel * in Paris is a*. In order to eliminate such results from the output we search the snippets returned for most frequent words preceding seeding term *Paris*.The resulting query has the form

$$\textit{"term is a *"}\ \text{-}qt_1 \ldots \text{-}qt_k,$$

where $qt_1 \ldots$-qt_k are qualificator terms exceeding some frequency threshold. Hypernym candidates are extracted from the snippets by taking all possible subsequences of words following the word matched by asterisk. This allow to extract right candidates from snippets like *Paris is a beautiful city*, or *Paris is a huge, often confusing city*.

3.2 Candidates Filtering

At this point one should decide that sequences of words form a term. Good terms should correspond to concepts. As it was mentioned in (Hearst, 1992), the measure of "termness likeliness" is domain-specific. For general domains terms should be as short as possible, while in biomedical applications terms can not be shortened because adjectives carry meaningful information. We use following criteria for termness likeliness estimation:

- relative difference between lengths of a term and the title of the nearest Wikipedia article (Wiki titles are polished by removing suffixes like - *Wikipedia, the free encyclopedia*);
- a logarithm of the number of pages containing the pattern *term is a **;
- a logarithm of the number of pages containing the pattern ** is a term*;
- and a logarithm of the number of pages containing *term* in their HTML titles.

The intuition behind these numbers is as follows. If there exist a Wikipedia article with exactly the same or very similar title it is likely that the term under consideration represents some concept. Number of pages containing term in their titles reflects importance of the concept. Finally, a term corresponding to a concept should appear in both part of IS-A relation.

The resulting likeliness value is a weighted sum of these numbers. Clearly, that absolute values of counting numbers are not important, because the number of occurrences of a IS-A pattern in the corpus depends of popularity of specific concept in the corpus. Let us consider an example:

term	wiki	title	* is a	is a *
bread	1.	12.9359	9.98525	10.6727
life	1.	15.6568	13.254	14.4916
sunny day	1.	10.1012	9.554	8.03496
semigroup	1.	6.20859	8.60594	7.76472
hypernym	0.	5.0876	7.73456	5.5835
fantastic destination	0.428571	4.39445	8.73311	2.89037
narrow garage	0.26087	5.54126	2.89037	0
garage	1.	13.9627	10.4113	10.5031
hesitation pause	0.625	1.79176	2.07944	1.79176
collaborative model	1.	6.10925	6.89264	5.15329
referential ambiguity	0.	3.43399	0	1.94591

One can find that if a term corresponds to some concept, then the ratio of last two columns is close to one, while for other terms this is not the case. Comparing terms *narrow garage* (which is not a

[2] According to search engine language the asterisk stands for arbitrary word. Putting it into the query forces the engine to include a word following *is a* into a snippet.

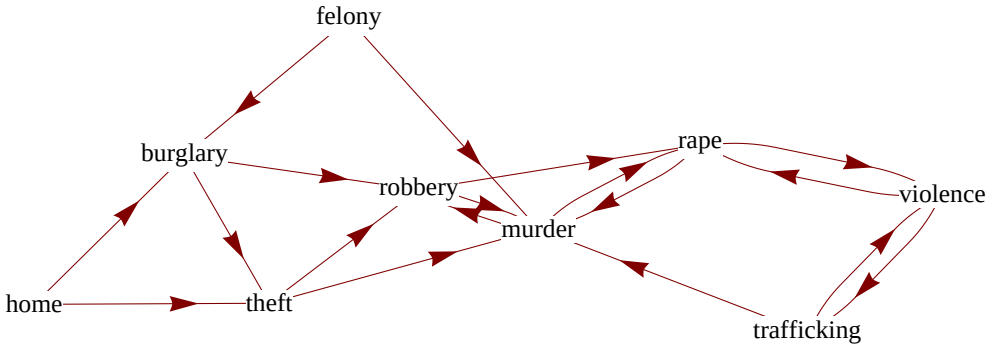

Figure 1: Felony is a crime.

concept) and *referential ambiguity* (a concept) one can assume that small values for number of * *is a* pages are allowed, and this is an indication of *bottom concepts*. Finally, let us note that term *sunny day* scores so high because of existence of a quite popular music band called "A sunny day in Glasgow".

The score of a term t is computed by a function $score(t) = f(w(t), u(t)/d(t))$, where $u(t)$ is the logarithm of number of * *is a* pages, $d(t)$ is the logarithm of number of *is a* * pages, that tries to assign large values to (1) wiki-terms, (2) terms with $u(t)/d(t)$ close to one, and (3) terms that have zero $u(t)/d(t)$ ratio.

For each hypernym candidate we compute its score and Part-Of-Speech tags (we use (Tsuruoka and Tsujii, 2005) [3]). A candidate passes the test if it either has a score greater some threshold, or it has a score greater than average and ends with a noun. Threshold value 0.5 guarantees that terms presented in wiki titles are not removed because of erroneous POS tagging.

3.3 Hypernym Validity Estimation

The goal of hypernym validity estimation is computing for given terms T and H a real number $\alpha \in [0, 1]$ reflecting the probability that H is a hypernym of T. The method is based on the assumption that *the set of hyponyms of a correct hypernym of the seeding term should contain terms that are semantically similar to the seeding term*. We perform the following steps.

1. Evaluate the query * *is a H* and extract from the snippets candidates for terms T neighbors (terms sharing with T common hypernym).
2. For selected candidates we construct a graph, similar to the *hypernym pattern linkage graph* described in (Kozareva *et al.*, 2008). Graph nodes correspond to T's neighbor candidates. Each node has exactly k out-going edges (in the examples below $k = 2$) leading to nodes with highest similarity measure. The similarity between two nodes, say N_1 and N_2, measured as the number of pages returned by the search engine for the query

   $$N_1, N_2 \textit{ and other Hs}.$$

   Due to search engine query limitations the actual query is N_1 *NEAR* N_2 *NEAR "and other Hs"*. Examples of similarity graphs for (T,H)={*felony*, *crime*}, and (T,H)={*robbery*, *crime*} represented on Fig. 1 and Fig. 2, respectively.
3. Finally, we apply PageRank algorithm to compute the weight of the node corresponding to the seeding term T.

[3] Available at http://www-tsujii.is.s.u-tokyo.ac.jp/ tsuruoka/postagger/

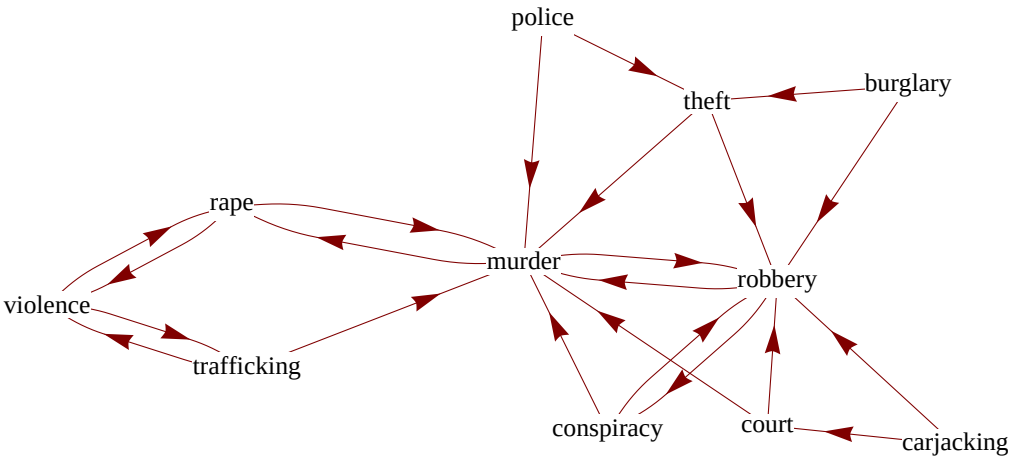

Figure 2: Robbery is a crime.

Let us note that we use an is-a query for T's hypernym candidates set construction and plural form *and other Hs* for validity estimation. It gives additional filter for hypernym candidates. If a term H was erroneously selected as a hypernym candidate for T, then it is likely that the "plural" query yields the empty set.

Usage of PageRank algorithm for validity estimation may be justified by the fact that it gives a higher value to a node if there exist incoming edges. Recall that incoming edge indicates that two term frequently occur in lexical pattern *A,B and other Cs*.

4 Experimental Results

Our experiment dials with the set of terms that are semantically related to the concept of *crime* in Roguet thesaurus: *robbery, theft, burglary, banditry, fraud, swindle, snatch, plunder, stealing, larceny, felony, identity theft, crime against humanity*.

The result of * *is a crime* query is the set (with number of patterns found):

murder	504	crime	6
burglary	62	conspiracy	6
theft	58	immigration	4
rape	45	offense	3
robbery	38	forgery	3
home	18	act	3
violence	16	evidence	2
misdemeanor	9	computer	2
court	9	terrorism	1
vandalism	8	clayton	1
law	8		

The output of the algorithm that estimates how likely that term crime is a hypernym of terms from the test set is:

fraud	0.830539	identity theft	0.896961
theft	0.852935	plunder	0.907692
felony	0.877171	burglary	0.943271
larceny	0.882474	crime against humanity	1.
robbery	0.885167	snatch	1.
banditry	0.888889	swindle	1.
stealing	0.888889		

For terms such as *joke*, that are definitely not related to crimes, the algorithm states outputs zero validity of a hypernym relation between joke and crime.

5 Conclusion

In this paper an approach to automated terminological hyponym/hypernum hierarchy construction using Internet search engine is proposed. The method combines various well-known techniques of lexical patterns (Hearst, 1992), such as doubly-anchored hyponyms search (Kozareva *et al.*, 2008). Very preliminary experimental results show that such combination could yield meaningful output without usage of any handcoded lexical resources.

There is a lot of work to do before one can state that this approach is feasible for real life applications. For example, a more sophisticated algorithms for hyponym construction can be used. In particular, a bootstrapping algorithm (Kozareva *et al.*, 2008) that starts from the seeding term and its hypernym and iteratively extends the set of term's candidates using the *A, B and other Cs* query. Although multi-word terms are very frequent in real text the problem of their identification was not addressed. Hypernym extraction from snippets like *Typhoon is a twin-engine canard-delta wing multirole aircraft* or *Typhoon is a fantastic aircraft* is a crucial feature. Finally, an extended evaluation on manually verified data should be performed.

References

Berland, Matthew and Eugene Charniak. 1999. Finding parts in very large corpora. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pp. 57–64, Morristown, NJ, USA. Association for Computational Linguistics.

Bollegala, Danushka, Yutaka Matsuo, and Mitsuru Ishizuka. 2009. A relational model of semantic similarity between words using automatically extracted lexical pattern clusters from the web. In *EMNLP '09: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pp. 803–812, Morristown, NJ, USA. Association for Computational Linguistics.

Chirita, Paul Alexandru, Claudiu S. Firan, and Wolfgang Nejdl. 2007. Personalized query expansion for the web. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 7–14, New York, NY, USA. ACM.

Cilibrasi, R. L. and P. M. B. Vitanyi. 2007. The google similarity distance. *IEEE Transactions on Knowledge and Data Engineering*, 19(3), 370–383.

Dang, Chenghua, Xinjun Luo, and Haibin Zhang. 2008. Wordnet-based summarization of unstructured document. *W. Trans. on Comp.*, 7(9), 1467–1472.

Drumond, Lucas and Rosario Girardi. 2010. Extracting ontology concept hierarchies from text using markov logic. In *SAC '10: Proceedings of the 2010 ACM Symposium on Applied Computing*, pp. 1354–1358, New York, NY, USA. ACM.

Girju, Roxana, Adriana Badulescu, and Dan Moldovan. 2003. Learning semantic constraints for the automatic discovery of part-whole relations. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pp. 1–8, Morristown, NJ, USA. Association for Computational Linguistics.

Gligorov, Risto, Zharko Aleksovski, Warner ten Kate, and F. van Harmelen. 2007. Using google distance to weight approximate ontology matches. In *Proceedings of the seventeenth World Wide Web conference WWWW'17*, pp. 767–775, Canada, May.

Harris, Z. S. 1968. *Mathematical Structures of Language*. Wiley, New York.

Hearst, Marti A. 1992. Automatic acquisition of hyponyms from large text corpora. In *In Proceedings of the 14th International Conference on Computational Linguistics*, pp. 539–545.

Jones, Rosie, Benjamin Rey, Omid Madani, and Wiley Greiner. 2006. Generating query substitutions. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pp. 387–396, New York, NY, USA. ACM.

Kozareva, Zornitsa, Ellen Riloff, and Eduard Hovy. 2008. Semantic class learning from the web with hyponym pattern linkage graphs. In *Proceedings of the Association for Computational Linguistics*.

Li, Jianqiang, Yu Zhao, and Bo Liu. 2009. Fully automatic text categorization by exploiting wordnet. In *AIRS '09: Proceedings of the 5th Asia Information Retrieval Symposium on Information Retrieval Technology*, pp. 1–12, Berlin, Heidelberg. Springer-Verlag.

Lopez, Vanessa, Victoria Uren, Enrico Motta, and Michele Pasin. 2007. Aqualog: An ontology-driven question answering system for organizational semantic intranets. *Web Semant.*, 5(2), 72–105.

Pantel, Patrick and Dekang Lin. 2002. Discovering word senses from text. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 613–619, New York, NY, USA. ACM Press.

Sanderson, Mark and Bruce Croft. 1999. Deriving concept hierarchies from text. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 206–213, New York, NY, USA. ACM.

Shinzato, Keiji and Kentaro Torisawa. 2004. Extracting hyponyms of prespecified hypernyms from itemizations and headings in web documents. In *COLING '04: Proceedings of the 20th international conference on Computational Linguistics*, p. 938, Morristown, NJ, USA. Association for Computational Linguistics.

Smith, Barry. 2004. Ontology. In *The Blackwell Guide to the Philosophy of Computing and Information*, pp. 155–166. Blackwell.

Tokunaga, Kosuke and Kentaro Torisawa. 2005. Automatic discovery of attribute words from web documents. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP-05), pages 106118, Jeju Island, Korea*, pp. 106–118.

Tsuruoka, Yoshimasa and Jun'ichi Tsujii. 2005. Bidirectional inference with the easiest-first strategy for tagging sequence data. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pp. 467–474, Morristown, NJ, USA. Association for Computational Linguistics.

van Heijst, Gertjan, A. Th. Schreiber, and Bob J. Wielinga. 1997. Using explicit ontologies in kbs development. *Int. J. Hum.-Comput. Stud.*, 46(2), 183–292.