

Extracting Semantic Relations Using Dependency Paths

Josh King and Venu Satuluri

Computer Science and Engineering Department

The Ohio State University

{kingjo, satuluri}@cse.ohio-state.edu

Abstract

Lexical semantic relations are useful in a variety of natural language applications, but to collect, update and maintain them by hand is tedious and costly. We experiment with the use of dependency paths, rather than regular expressions, as the formalism for representing patterns that may be indicative of a semantic relation between noun pairs. Our corpus is Wikipedia article abstracts, and we tag our training set using WordNet. We collect frequent dependency paths and using these as features, train two classification algorithms to predict the existence of a semantic relation among noun pairs in the test corpus, which predictions we evaluate using human judgment as the gold standard. Our experiments with hypernymy were successful; those with meronymy less so.

1 Introduction

Lexical semantic relations encode knowledge about the world (common-sense or otherwise) and hence are essential for many natural language applications such as question-answering, natural language understanding and information retrieval. Semantic relations between noun pairs include hyponymy (a cat is a type of animal, so “cat” is a hyponym of “animal”), hypernymy (the opposite of hyponymy), meronymy (“wing” is a meronym of “airplane” in that a wing is part of an airplane), and antonymy (“happy” is the opposite of “sad”). For an example of the usefulness of semantic relations, a program that is aware of the relation *hypernym*(mammal, human) will be able to infer from the proposition “All mammals are warm-blooded” the new proposition “Humans are warm-blooded”.

WordNet (Fellbaum, 1998) is a database of words which also includes lexical semantic relations, but was hand built by a slow and tedious process, and

furthermore, does not have thorough coverage¹ of domain-specific semantic relations. To tackle these problems with hand-built databases of lexical semantic relations, there has been much interest in methods for automatically finding such relations. In this paper we implement a part of the work explained in (Snow et al., 2005), whose primary goal is the discovery of hypernyms for use in building a taxonomy (Snow et al., 2006). The approach taken in (Snow et al., 2005) is to use dependency paths as features for prediction of hypernymy, and later to use the probability of nouns being co-ordinate terms to enhance the probability of there being a hypernymy relation. We investigate only the usage of dependency paths as features and do not calculate probabilities of nouns being co-ordinate terms in this paper. Whereas (Snow et al., 2005) is interested in the presence of hypernym relationships between nounpairs for the purpose of ontology building, we believe that their method should be useful for discovering noun pairs which share other semantic relations; to that end, we use their method to attempt the discovery of meronyms as well.

In section 2 we explain the paper’s use of dependency paths, and also briefly discuss the two machine learning algorithms we used - Logistic Regression and Naive Bayes Classification. In section 3 we explain our method and experiments and in section 4 we explain our results and evaluation, both qualitative and quantitative. We mention future work to be done and conclude in section 6.

2 Background and Relevant Prior Work

(Hearst, 1992) was the first to propose the idea that lexical patterns found in plain text can be used to automatically infer semantic relations. (Hearst, 1992) used the example (initially from Grolier’s Encyclopedia),

¹This is one impetus for (Snow et al., 2006) – their product can supplement WordNet’s sparsity.

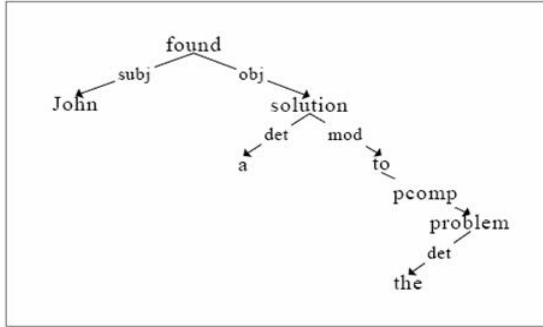


Figure 1: A dependency parse of the sentence ‘John found a solution to the problem’. The labels of the edges represent the dependency relation. (Lin and Pantel, 2001)

The bow lute, such as the Bambara ndang, is plucked and has an individual curved neck for each string.

to argue that for a native speaker of English who had never before encountered the term Bambara ndang, it only takes some small mental work to infer that the Bambara ndang is a type of bow lute. The key insight is that it is possible to identify phrase templates or patterns that are highly indicative of a semantic relationship between the words involved. In the case above the pattern ‘X such as Y’ indicates that X is a hypernym of Y, *i.e.*, in the example, “bow lute” is a hypernym of “Bambara ndang”. There has been much work on extracting semantic relations and named entity recognition using variants of this idea (*e.g.*, (Etzioni et al., 2005)).

(Snow et al., 2005) proposed the idea of representing such phrase templates using the more general representation of *dependency paths*; dependency paths had been successfully used in the past for applications such as question-answering (Lin and Pantel, 2001). A dependency parse of a sentence produces a dependency tree, with the nodes representing words in the sentence and edges representing the dependency relations between nodes. Figure 1 shows an example of a dependency tree representation of the sentence ‘John found a solution to the problem’ (Lin and Pantel, 2001). We use the textual representation described in (Snow et al., 2005) for an edge: each edge is represented as a tuple of the form: $(word_1, CATEGORY_1, RELATION, CATEGORY_2, word_2)$. The edge defines a RELATION between $word_1$ and $word_2$, which are of grammatical types $CATEGORY_1$ and $CATEGORY_2$, respectively. The

dependency path between any pair of words can then be specified as a list of such edge tuples.

For an illustration of how dependency paths are more powerful than plain lexical patterns, consider the two sentences ‘EU countries like France have been raising their taxes recently’ and ‘Many Americans like France and want to visit it’. “country” is a hypernym of “France” in the first sentence, but “American” is not a hypernym of “France” in the second. Dependency paths catch this distinction², whereas the regular expression ‘X like Y’ does not. For an example of how dependency paths are also more general than lexical patterns, consider that all the lexical patterns ‘X is a Y’, ‘X is a small Y’, ‘X is an American Y’, etc., can be represented by the same dependency path ‘N:SUBJ:N’.

2.1 Extended Dependency Paths

We implement two extensions to dependency paths specified in (Snow et al., 2005) that provides more capturing power. (Snow et al., 2005) showed the example in Figure 2, which illuminates the two extensions using the partial dependency path of the phrase ‘such authors as Herrick and Shakespeare’.

- Distribution of dependencies across conjunctions, *i.e.*, all the words in a conjunction use the same dependency path to a noun not in the conjunction. In figure 2, “authors” is linked to “Herrick” via the dependency path ‘N:MOD:PREP:as, as:PREP:PCOMP-N:N’, so “authors” should also be linked to “Shakespeare” with the same path, rather than by appending the additional edge ‘N:CONJ:N’.
- Addition of optional *satellite links* which extend from the noun pairs being bridged and were not part of the original dependency path. This is done because there may be some edges which are not strictly part of the dependency path but are vital to provide reliable evidence of hypernymy. These satellite links made it possible for dependency paths to express the lexical patterns that (Hearst, 1992) originally proposed as good candidates for extracting evidence of hypernymy. The dependency paths with satellite links extracted from the phrase whose partial dependency tree is shown in Figure 2 are shown in Table 1.

²‘+N:MOD:PREP:like,like:PCOMP-N:N’ in the first sentence versus ‘+N:S:V:like,like:V:OBJ:N’ in the second

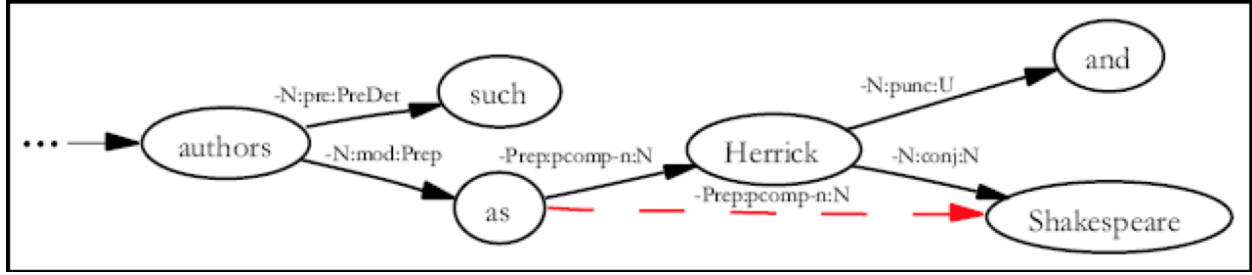


Figure 2: MINIPAR dependency tree example, with transform, for ‘such authors as Herrick and Shakespeare’ (Snow et al., 2005)

Dependency Paths with Optional Satellite Links for (author,Herrick)
N:MOD:PREP:as,as:PREP:PCOMP-N:N
N:MOD:PREP:as,as:PREP:PCOMP-N:N(N:PUNC:U:and)
(N:PRE:PREDET:such)N:MOD:PREP:as,as:PREP:PCOMP-N:N
(N:PRE:PREDET:such)N:MOD:PREP:as,as:PREP:PCOMP-N:N(N:PUNC:U:and)

Table 1: Dependency paths with optional satellite links for the partial dependency tree shown in Figure 2, for the noun pair (author,Herrick)

As in (Snow et al., 2005), all nouns in the dependency path are removed to make the pattern more general.

2.2 Learning Algorithms

We use two machine learning algorithms – Naive Bayes classification and Logistic Regression – to learn the semantic relations (see section 3 for an explanation of the feature vectors used in these algorithms).

2.3 Naive Bayes classification

In this method, we calculate $P(f_i|c_j)$ for each feature vector f_i and class c_j during the training phase. We then calculate $P(c_j|f_1,\dots,f_n)$ for the probability that a test instance belongs to class c_j using Bayes’ rule and assuming independence between the various features.

$$(1) \quad P(c_j|F) = P(c_j) \prod_i P(f_i|c_j)$$

We chose Naive Bayes because it is easy to implement, the parameters lend themselves to interpretation and because it produced good results for (Snow et al., 2005).

2.4 Logistic Regression

Logistic regression is a model in which the logarithm of the odds of a class of an instance is modeled

as a linear function of the features of that instance. Equivalently, given a feature vector F , the posterior probability for a class c_j can be written as

$$(2) \quad P(c_j|F) = g\left(\sum_i w_i f_i\right),$$

where $g(z) = \frac{1}{1+e^{-z}}$. There’s an additional bias feature added to the above feature vector with value always equal to 1, which is meant to take care of the intercept. These weights represent the parameters of the LR system. We chose LR as one of our algorithms because, again, it produced good results for (Snow et al., 2005), and because LR is capable of modeling high-dimensional data where many of the features may be irrelevant.

We use the program for logistic regression written by Paul Komarek and Andrew Moore³. This program learns the LR parameters through Truncated Regularized IRLS (Iteratively Re-weighted Least Squares), which optimizes likelihood using a linear Conjugate Gradient algorithm that runs faster than non-linear alternatives and has been shown to be at least as accurate (Komarek and Moore, 2005).

3 Method

We use a method similar to what (Snow et al., 2005) describe to extract noun pairs exhibiting a semantic

³Downloaded from <http://www.autonlab.org>

relationship. We use supervised machine learning to find dependency paths between noun pairs which may be indicative of a semantic relation. The steps are as follows:

- Run the broad coverage dependency parser MINIPAR to extract dependency paths between any noun pair in a sentence. All dependency paths that occur between at least five unique noun pairs are added to our collection of patterns to be later used as features for the classification.
- Use WordNet to tag a noun pair as either known to be exhibiting the relation in question or known to be not exhibiting the relation. Known to be not exhibiting the relation means WordNet contains both words, but they are not connected by the semantic relation we are examining. Noun pairs which WordNet does not know about are not included in our training set.
- For every noun pair in our corpora, we compile a feature vector containing frequency counts for how often it was found using the particular dependency path. That is, the i -th value of the feature vector is the number of times the i -th dependency path was found linking the noun pair in the corpora.
- Train classifiers using the training set of WordNet tagged noun pairs.
- Test: collect a feature vector for every noun pair in our test corpora, using the same features (dependency paths in the same order) collected during training; use the classifier to determine whether the noun pair exhibits the semantic relation we've trained for.

3.1 Experimental Setup

Our training corpora was a subset of Wikipedia abstracts (Wikipedia, 2006). Not all the abstracts in Wikipedia are well-formed, i.e., some may be empty, some may only have special characters, e.g. the spelling of Greek names using the Greek alphabet, so we try to only keep abstracts that are mostly well-formed by removing unknown characters and not parsing an abstract if it is smaller than some threshold length - most, if not all of the letters being unknown characters would produce this effect. We then bucket the abstracts into 75 files so that the i -th abstract in Wikipedia is in the $(i \bmod 75)$ -th file. In this way we ensure that each file has a roughly

independent and identical distribution over the entries as any other file and is a subset of the original Wikipedia articles such that its size does not exceed $1/75$ -th of the original. Such manipulations of the data were performed to keep the experiment tractable with our computational power.

When we parsed this data using MINIPAR (Lin, 1998), we collected a total of 57,903 dependency paths to use as features. (This is comparable to the number of dependency paths (Snow et al., 2005) found.) We also collected 403,000 noun pairs for our training set and tagged them for hypernymy and meronymy separately and automatically using WordNet⁴. There were 4,575 and 434 positive examples for hypernymy and meronymy respectively. We built the feature vectors for each noun pair encountered in the same sentence in our training data and which were within three edges of each other in the dependency tree and trained the two classifiers - Naive Bayes and Logistic Regression - on this data.

We then tested the algorithm against decisions made by two human judges on 200 noun pairs selected at random from one of the reduced abstract subsets that were not trained on.

Table 2 shows the operation of the method for an example sentence from our test corpus, 'RBI is a common baseball term for runs batted in.' and the noun pair contained in that sentence (RBI,term). The base pattern, 'N:SUBJ:N', is repeated in three patterns that also contain the optional satellite links around the second word, 'term'. The feature vector for the noun pair is the sparse vector with only those indices shown containing non-zero values. The feature vector encodes a good pattern for detecting hypernymy, which is the dependency of the second word to the determiner 'a', the adjective 'common', and a be verb, and the first word is the subject of the sentence. For the noun pair (RBI,term) to be tagged by our classifier as exhibiting hypernymy, the threshold of acceptance now only needs to be set below the values shown for either classifier.

4 Evaluation

Our test corpus was one of the $1/75$ -th portions of our corpus, described in section 3.1. We first built the feature vectors for all the noun pairs occurring in the same sentence in the test corpus, and ran our classification algorithms to get, for each noun pair,

⁴We used Jason Rennie's WordNet::QueryData Perl interface to WordNet-2.1, downloadable from <http://search.cpan.org/dist/WordNet-QueryData/>

sentence	RBI is a common baseball term for runs batted in.
pattern ₄	N:SUBJ:N
pattern ₃	(BE:VBE:+PRED:N),N:SUBJ:N
pattern ₉	(A:DET:DET:N),N:SUBJ:N
pattern ₂₂₁₉	(COMMON:A:MOD:N),N:SUBJ:N
feature vector	{(3,1),(4,1),(9,1),(2219,1)}
Logistic Regression Score	0.1673
Naive Bayes Score	0.0130

Table 2: Determining whether the noun pair (RBI,term) has a hypernymy relation.

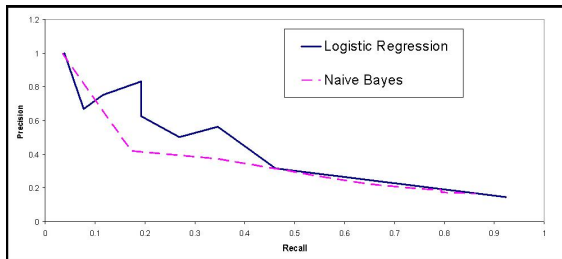


Figure 3: Precision-recall curves of Hypernym classifiers on test set

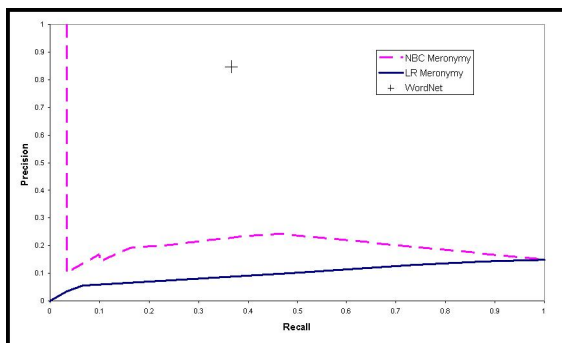


Figure 4: Precision-recall curves of Meronym classifiers on test set

the probability that there exists the semantic relation we are looking for between that noun pair. 200 noun pairs were randomly picked from the test corpus and then labeled by human annotators (the authors) independently as being either positive or negative instances of the semantic relation. We then used those noun pairs which there was agreement among the annotators as the gold standard against which to evaluate our algorithms. Here we explain our results for the two experiments we performed - one for hypernymy and the other for meronymy.

4.1 Results for Hypernymy

Among the test set of 200 random noun pairs, human annotators agreed upon 30 noun pairs as being in a hypernym relationship, whereas there was disagreement about 24 noun pairs. Out of those 30 noun pairs, only 4 were tagged as positive examples, thus indicating the sparse coverage of WordNet (it must be kept in mind that we considered instances also to be hyponyms). Figure 3 shows the performance of the two classifiers with a recall and precision plot obtained by varying the threshold for decisions. We show the F-scores for WordNet and the two best classifiers in Table 4.1. The best F-score of our Logistic Regression algorithm (0.4286) is better than that reported by (Snow et al., 2005) (0.3592). We hypothesize two reasons for this:

- (Snow et al., 2005) hand-labeled a test set of 5,387 noun pairs from random sentences, which is much larger than our own test set of 200 noun pairs.
- Wikipedia article abstracts may be particularly well suited for extraction of semantic relations, because an abstract should have the most relevant material about a concept given in a concise fashion. In particular, it might be a better choice for training corpus than either Wikipedia full articles, which contain more information about a concept than is necessary to infer semantic relations, or Newswire corpora, which may contain evidential patterns with greater rarity.

Some of the high-scoring patterns discovered by our algorithm are shown in Table 4. Note that the patterns are essentially dependency path representations of the Hearst patterns, but none of the example sentences would have been caught by lexical pattern matching.

Interannotator Agreement:	0.8844
Best Naive Bayesian Classifier:	0.3596
Best Logistic Regression:	0.4286
Wordnet:	0.2312
Best Hypernym Classifier in (Snow et al., 2005):	0.3592

Table 3: F-scores of hypernym classifiers on hand-labeled test set

Dependency path	Example Sentence
NP1 N:SUBJ:N NP2	<u>RBI</u> is a common baseball <u>term</u> for runs batted in.
NP1 N:CONJ:N, (OTHER:A:MOD:N) NP2	The film was produced by Paramount Pictures ... who also produced <u>The Mummy</u> , <u>Tombstone</u> and 21 other <u>feature films</u> .
NP1 (U:PUNC:N), N:+NN:N, N:+S:VBE:BE, BE:VBE:PRED:N NP2	The <u>tomatillo</u> (<i>Physalis ixocarp</i> or <i>Physalis philadelphica</i>) is a small, spherical and green or green-purple <u>fruit</u> surrounded by a paper-like husk formed from the calyx.

Table 4: Some high-scoring dependency paths discovered by our learning algorithm, with examples of sentences in which they have occurred. The underlined words represent the NP1 and the NP2 in the paths.

4.2 Results for Meronymy

For producing the test set of noun pairs for meronymy, we weighted noun pairs for which WordNet had the meronymy relation listed slightly more heavily than other noun pairs, as we thought the natural frequency of noun pairs that are meronyms is lower than that of hypernyms, and we did not want to label large numbers of noun pairs to obtain a significant number of positive examples. Consequently, among the test set of 200 noun pairs, human annotators agreed that 30 were positive examples, among which 13 were tagged as positive by WordNet. Our classification algorithms performed poorly, with the precision recall curves sloping upwards at an odd angle rather than downwards as expected, indicating that there was a strong random component to the probabilities output. We believe this may be because of two reasons:

- We had a small number of positive examples - 434 as compared to 4575 for Hypernymy - with which to train.
- Meronyms of an object may not have very good patterns that indicate the semantic relationship without a high false-positive rate. A qualitative analysis of the patterns that bridge some positive examples indicates this. For example, the pattern ‘X’s Y’ is often observed among noun pairs with this relation, but this

pattern also has a high false positive rate, *i.e.*, many noun pairs which are not in a meronymic relationship instantiate this pattern (e.g. ‘I borrowed John’s book’).

5 Grounds for improvement

The main area in which our algorithm is incomplete is the lack of word sense disambiguation. We may want to know which particular senses of the two words are in the semantic relationship, and this can be readily seen in examples such as “character” being a meronym of “file”, in a Computer Science sense, and also being a meronym of “play”, in a theatrical sense.

Another limitation of pattern based methods for semantic relation extraction in general is the problem of data sparsity, as noted by (Snow et al., 2005). This is because many words that do share a semantic relation may never appear in a sentence together in our corpus. We speculate that this problem can be partially alleviated by using dictionary-like corpora (such as Wikipedia article abstracts) which have a much higher coverage of concepts.

6 Conclusion

In this paper we implemented a method of extracting semantic relations using dependency paths, and our results show the power that dependency paths have to extract semantic relations. Our results are

Interannotator Agreement Meronymy	.91
Best Naive Bayesian Classifier	0.3182
Best Logistic Regression	0.2609
Wordnet Meronymy	0.5116

Table 5: F-score of meronym classifiers

slightly better than the results obtained in previous work, which we hypothesize was due to the dictionary-like nature of our data set (Wikipedia abstracts). We plan to examine this hypothesis in the future by seeing how well the algorithm performs on Wikipedia entries as a whole. Also, our results at the extraction of the meronym semantic relation were not as good as we were expecting. We plan to examine other relations as well as reexamining the meronym relation to further pursue whether this method can be expanded beyond hypernym extraction.

References

- Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence*.
- Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.
- Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*, Nantes, France.
- Paul Komarek and Andrew Moore. 2005. Making logistic regression a core data mining tool with tr-irls. In *Proceedings of the 5th International Conference on Data Mining Machine Learning*, page 4.
- Dekang Lin and Patrick Pantel. 2001. Discovery of inference rules for question-answering. *Natural Language Engineering*, 7:343–360.
- Dekang Lin. 1998. Dependency-based evaluation of minipar. In *Workshop at LREC'98 on The Evaluation of Parsing Systems*, Granada, Spain.
- Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. *NIPS 17, 2005*.
- Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2006. Semantic taxonomy induction from heterogeneous evidence. In *COLING/ACL 2006*.
- Wikipedia. 2006. Wikipedia, the free encyclopedia. <http://download.wikimedia.org> [Online; accessed 04-November-2006].