Taxonomy learning – factoring the structure of a taxonomy into a semantic classification decision

Viktor PEKAR
Bashkir State University
Ufa, Russia, 450000
vpekar@ufanet.ru

Steffen STAAB
Institute AIFB, University of Karlsruhe
http://www.aifb.uni-karlsruhe.de/WBS
& Learning Lab Lower Saxony
http://www.learninglab.de

Abstract

The paper examines different possibilities to take advantage of the taxonomic organization of a thesaurus to improve the accuracy of classifying new words into its classes. The results of the study demonstrate that taxonomic similarity between nearest neighbors, in addition to their distributional similarity to the new word, may be useful evidence on which classification decision can be based.

1. Introduction

Machine-readable thesauri are now an indispensable part for a wide range of NLP applications such as information extraction or semantics-sensitive information retrieval. Since their manual construction is very expensive, a lot of recent NLP research has been aiming to develop ways to automatically acquire lexical knowledge from corpus data.

In this paper we address the problem of large-scale augmenting a thesaurus with new lexical items. The specifics of the task are a big number of classes into which new words need to be classified and hence a lot of poorly predictable semantic distinctions that have to be taken into account. For this reason, knowledge-poor approaches such as the distributional approach are particularly suited for this task. Its previous applications (e.g., Grefenstette 1993, Hearst and Schuetze 1993, Takunaga et al 1997, Lin 1998, Caraballo 1999) demonstrated that cooccurrence statistics on a target word is often sufficient for its automatical classification into one of numerous classes such as synsets of WordNet.

Distributional techniques, however, are poorly applicable to rare words, i.e., those words for which a corpus does not contain enough cooccurrence data to judge about their meaning. Such words are the primary concern of many practical NLP applications: as a rule, they are semantically focused words and carry a lot of important information. If one has to do with a specific domain of lexicon, sparse data is a problem particularly difficult to overcome.

The major challenge for the application of the distributional approach in this area is, therefore, the development of ways to minimize the amount of corpus data required to successfully carry out a task. In this study we focus on optimization possibilities of an important phase in the process of automatically augmenting a thesaurus – the classification algorithm. The main hypothesis we test here is that the accuracy of semantic classification may be improved by taking advantage of information about taxonomic relations between word classes contained in a thesaurus.

On the example of a domain-specific thesaurus we compare the performance of three state-of-the-art classifiers which presume flat organization of thesaurus classes and two classification algorithms, which make use of taxonomic organization of the thesaurus: the "tree descending" and the "tree ascending" algorithms. We find that a version of the tree ascending algorithm, though not improving on other methods overall, is much better at choosing a superconcept for the correct class of the new word. We then propose to use this algorithm to first narrow down the search space and then apply the kNN method to determine the correct class among fewer candidates.

The paper is organized as follows. Sections 2 and 3 describe the classification algorithms under study. Section 4 describes the settings and data of the experiments. Section 5 details the evaluation method. Section 6 presents the results of the experiments. Section 7 concludes.

2. Classification methods

Classification techniques previously applied to distributional data can be summarized according to the following methods: the *k* nearest neighbor (kNN) method, the category-based method and the centroid-based method. They all operate on vector-based semantic representations, which describe the meaning of a word of interest (target word) in terms of counts[1] of its coocurrence with context words, i.e., words appearing within some delineation around the target word. The key differences between the methods stem from different underlying ideas about how a semantic class of words is represented, i.e. how it is derived from the original cooccurrence counts, and, correspondingly, what defines membership in a class.

The kNN method is based on the assumption that membership in a class is defined by the new instance's similarity to one or more individual members of the class. Thereby, similarity is defined by a similarity score as, for instance, by the cosine between cooccurrence vectors. To classify a new instance, one determines the set of *k* training instances that are most similar to the new instance. The new instance is assigned to the class that has the biggest number of its members in the set of nearest neighbors. In addition, the classification decision can be based on the similarity measure between the new instance and its neighbors: each neighbor may vote for its class with a weight proportional to its closeness to the new instance. When the method is applied to augment a thesaurus, a class of training instances is typically taken to be constituted by words belonging to the same synonym set, i.e. lexicalizing the same concept (e.g., Hearst and Schuetze 1993). A new word is assigned to that synonym set that has the biggest number of its members among nearest neighbors.

[1] Or, probabilities determined via Maximum Likelihood Estimation.

The major disadvantage of the kNN method that is often pointed out is that it involves significant computational expenses to calculate similarity between the new instance and every instance of the training set. A less expensive alternative is the category-based method (e.g., Resnik 1992). Here the assumption is that membership in a class is defined by the closeness of the new item to a generalized representation of the class. The generalized representation is built by adding up all the vectors constituting a class and normalising the resulting vector to unit length, thus computing a probabilistic vector representing the class. To determine the class of a new word, its unit vector is compared to each class vector. Thus the number of calculations is reduced to the number of classes. Thereby, a class representation may be derived from a set of vectors corresponding to one synonym set (as is done by Takunaga et al. 1997) or a set of vectors corresponding to a synonym set and some or all subordinate synonym sets (Resnik 1992).

Another way to prepare a representation of a word class is what may be called the centroid-based approach (e.g., Pereira et al. 1993). It is almost exactly like the category-based method, the only difference being that a class vector is computed slightly differently. All *n* vectors corresponding to class members are added up and the resulting vector is divided by *n* to compute the centroid between the *n* vectors.

3. Making use of the structure of the thesaurus

The classification methods described above presuppose that semantic classes being augmented exist independently of each other. For most existing thesauri this is not the case: they typically encode taxonomic relations between word classes. It seems worthwhile to employ this information to enhance the performance of the classifiers.

3.1 Tree descending algorithm

One way to factor the taxonomic information into the classification decision is to employ the "tree-descending" classification algorithm, which is a familiar technique in text categorization. The principle behind this approach is that the semantics of every concept in the thesaurus tree retains some of the semantics of all its hyponyms in such a way that the upper the concept, the more relevant semantic characteristics of its hyponyms it reflects. It is thus feasible to determine the class of a new word by descending the tree from the root down to a leaf. The semantics of concepts in the thesaurus tree can be represented by means of one of the three methods to represent a class described in Section 2. At every tree node, the decision which path to follow is made by choosing the child concept that has the biggest distributional similarity to the new word. After the search has reached a leaf, the new word is assigned to that synonym set, which lexicalizes the concept that is most similar to the new word. This manner of search offers two advantages. First, it allows to gradually narrow down the search space and thus save on computational expenses. Second, it ensures that, in a classification decision, more relevant semantic distinctions of potential classes are given more preference than less relevant ones. As in the case with the category-based and the centroid-based representations, the performance of the method may be greatly dependent on the number of subordinate synonyms sets included to represent a concept.

3.2 Tree ascending algorithm

Another way to use information about inter-class relations contained in a thesaurus is to base the classification decision on the combined measures of distributional similarity and taxonomic similarity (i.e., semantic similarity induced from the relative position of the words in the thesaurus) between nearest neighbors. Suppose words in the nearest neighbors set for a given new word, e.g., *trailer*, all belong to different classes as in the following classification scenario: *box* (similarity score to *trailer*: 0.8), *house* (0.7), *barn* (0.6), *villa* (0.5) (Figure 1). In this case, kNN will classify *trailer* into the class CONTAINER, since it appears to have biggest similarity to *box*. However, it is obvious that the most likely class of *trailer* is in a different part of the thesaurus: in the nearest neighbors set there are three words which, though not belonging to one class, are semantically close to each other. It would thus be safer to assign the new word to a concept that subsumes one or all of the three semantically similar neighbors. For example, the concepts DWELLING or BUILDING could be feasible candidates in this situation.

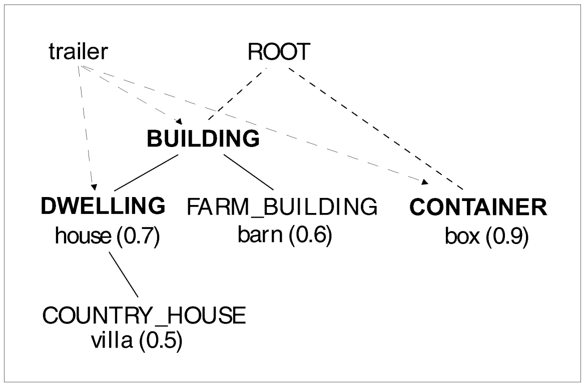

Figure 1. A semantic classification scenario.

The crucial question here is how to calculate the total of votes for these two concepts to be able to decide which of them to choose or whether to prefer CONTAINER. Clearly, one cannot sum or average the distributional similarity measures of neighbors below a candidate concept. In the first case the root will always be the best-scoring concept. In the second case the score of the candidate concept will always be smaller than the score of its biggest-scoring hyponym.

We propose to estimate the total of votes for such candidate concepts based on taxonomic similarity between relevant nodes. The taxonomic similarity between two concepts is measured according to the procedure elaborated in (Maedche & Staab, 2000). Assuming that a taxonomy is given as a tree with a set of nodes N, a set of edges $E \subset N \times N$, a unique root ROOT $\in N$, one first determines the least common superconcept of a pair of concepts a,b being compared. It is defined by

$$lcs(a,b) = \arg\min_{c \in N} (\delta(a,c) + \delta(b,c) + \delta(root,c)) \quad (1)$$

where $\delta(a,b)$ describes the number of edges on the shortest path between a and b. The taxonomic similarity between a and b is then given by

$$T(a,b) = \frac{\delta(root,c)}{\delta(a,c) + \delta(b,c) + \delta(root,c)} \quad (2)$$

where c = lcs(a,b). T is such that $0 \le T \le 1$, with 1 standing for the maximum taxonomic similarity. T is directly proportional to the number of edges from the least common superconcept to the root, which agrees with the intuition that a given number of edges between two concrete concepts signifies greater similarity than the same number of edges between two abstract concepts.

We calculate the total of votes for a candidate concept by summing the distributional similarity measures of its hyponyms to the target word *t* each weighted by the taxonomic similarity measure between the hyponym and the candidate node:

$$W(n) = \sum_{h \in I_n} sim(t,h) \cdot T(n,h) \qquad (3)$$

where I_n is the set of hyponyms below the candidate concept *n*, *sim*(*t*,*h*) is the distributional similarity between a hyponym *h* and the word to be classified *t*, and *T(n,h)* is the taxonomic similarity between the candidate concept and the hyponym *h*.

4. Data and settings of the experiments

The machine-readable thesaurus we used in this study was derived from GETESS[2], an ontology for the tourism domain. Each concept in the ontology is associated with one lexical item, which expresses this concept. From this ontology, word classes were derived in the following manner. A class was formed by words lexicalizing all child concepts of a given concept. For example, the concept CULTURAL_EVENT in the ontology has successor concepts PERFORMANCE, OPERA, FESTIVAL, associated with words *performance*, *opera*, *festival* correspondingly. Though these words are not synonyms in the traditional sense, they are taken to constitute one semantic class, since out of all words of the ontology's lexicon their meanings are closest. The thesaurus thus derived contained 1052 words and phrases (the corpus used in the study had data on 756 of them). Out of the 756 concepts, 182 were non-final; correspondingly, 182 word classes were formed. The average depth level of the thesaurus is 5.615, the maximum number of levels is 9. The corpus from which distributional data was obtained was extracted from a web site advertising hotels around the world[3]. It contained around 1 million words.

Collection of distributional data was carried out in the following settings. The preprocessing of corpus included a very simple stemming (most common inflections were chopped off; irregular forms of verbs, adjectives and nouns were changed to their first forms). The context of usage was delineated by a window of 3 words on either side of the target word, without transgressing sentence boundaries. In case a stop word other than a proper noun appeared inside the window, the window was accordingly expanded. The stoplist included 50 most frequent words of the British National Corpus, words listed as function words in the BNC, and proper nouns not appearing in the sentence-initial position. The obtained frequencies of cooccurrence were weighted by the 1+log weight function. The distributional similarity was measured by means of three different similarity measures: the Jaccard's coefficient, L1 distance, and the skew divergence. This choice of similarity measures was motivated by results of studies by (Levy et al 1998) and (Lee 1999) which compared several well known measures on similar tasks and found these three to be superior to many others. Another reason for this choice is that there are different ideas underlying these measures: while the Jaccard's coefficient is a binary measure, L1 and the skew divergence are probabilistic, the former being geometrically motivated and the latter being a version of the information theoretic Kullback Leibler divergence (cf., Lee 1999).

5. Evaluation method

The performance of the algorithms was assessed in the following manner. For each algorithm, we held out a single word of the thesaurus as the test case, and trained the system on the remaining 755 words. We then tested the algorithm on the held-out vector, observing if the assigned class for that word coincided with its original class in the thesaurus, and counting the number of correct classifications ("direct hits"). This was repeated for each of the words of the thesaurus.

However, given the intuition that a semantic classification may not be simply either right or wrong, but rather of varying degrees of appropriateness, we believe that a clearer idea about the quality of the classifiers would be given by an evaluation method that takes into account "near misses" as well. We therefore evaluated the performance of the algorithms also in terms

[2] http://www.daml.org/ontologies/171
[3] http://www.placestostay.com

of Learning Accuracy (Hahn & Schnattinger 1998), i.e., in terms of how close on average the proposed class for a test word was to the correct class. For this purpose the taxonomic similarity between the assigned and the correct classes is measured so that the appropriateness of a particular classification is estimated on a scale between 0 and 1, with 1 signifying assignment to the correct class. Thus Learning Accuracy is compatible with the counting of direct hits, which, as will be shown later, may be useful for evaluating the methods.

In the following, the evaluation of the classification algorithms is reported both in terms of the average of direct hits and Learning Accuracy ("direct+near hits") over all words in the thesaurus.

To have a benchmark for evaluation of the algorithms, a baseline was calculated, which was the average hit value a given word gets, when its class label is chosen at random. The baseline for direct hits was estimated at 0.012; for direct+near hits, it was 0.15741.

6. Results

We first conducted experiments evaluating performance of the three standard classifiers. To determine the best version for each particular classifier, only those parameters were varied that, as described above, we deemed to be critical in the setting of thesaurus augmentation.

In order to get a view on how the accuracy of the algorithms was related to the amount of available distributional data on the target word, all words of the thesaurus were divided into three groups depending on the amount corpus data available on them (see Table 1). The amount of distributional data for a word (the "frequency" in the left column) is the total of frequencies of its context words.

Table 1. Distribution of words of the thesaurus into frequency ranges

Frequency range	# words in the range
0-40	274
40-500	190
>500	292

The results of the evaluation of the methods are summarized in the tables below. Rows specify the measures used to determine distributional similarity (JC for Jaccard's coefficient, L1 for the L1 distance and SD for the skew divergence) and columns specify frequency ranges. Each cell describes the average of direct+near hits / the average of direct hits over words of a particular frequency range and over all words of the thesaurus. The statistical significance of the results was measured in terms of the one-tailed chi-square test.

kNN. Evaluation of the method was conducted with *k*=1, 3, 5, 7, 10, 15, 20, 25, and 30. The accuracy of classifications increased with the increase of *k*. However, starting with *k*=15 the increase of *k* yielded only insignificant improvement. Table 2 describes results of evaluation of kNN using 30 nearest neighbors, which was found to be the best version of kNN.

Table 2. kNN, *k*=30.

	0-40	40-500	>500	Overall
JC	.33773 /.17142	.33924 /.15384	.40181 /.12457	.37044 /.15211
L1	.33503 /.16428	.38424 /.21025	.38987 /.14471	.37636 /.17195
SD	.31505 /.14285	.36316 /.18461	.45234 /.17845	.38806 /.17063

Category-based method. To determine the best version of this method, we experimented with the number of levels of hyponyms below a concept that were used to build a class vector). The best results were achieved when a class was represented by data from its hyponyms at most three levels below it (Table 3).

Table 3. Category-based method, 3 levels

	0-40	40-500	>500	Overall
JC	.26918 /.12142	.34743 /.17948	.47404 /.28282	.37554 /.2023
L1	.27533 /.125	.41736 /.25128	.56711 /.38383	.43242 /.26190
SD	.28589 /.12857	.34932 /.18461	.51306 /.31649	.39755 /.21957

Centroid-based method. As in the case with the category-based method, we varied the number of levels of hyponyms below the candidate concept. Table 4 details results of evaluation of the best version of this method (a class is represented by 3 levels of its hyponyms).

Table 4. Centroid-based method, 3 levels.

	0-40	40-500	>500	Overall
JC	.17362 /.07831	.18063 /.08119	.30246 /.14434	.22973 /.10714
L1	.21711 /.09793	.30955 /.13938	.37411 /.1687	.30723 /.12698
SD	.22108 /.09972	.23814 /.11374	.36486 /.16147	.28665 /.10714

Comparing the three algorithms we see that overall, kNN and the category-based method exhibit comparable performance (with the exception of measuring similarity by L1 distance, when the category-based method outperforms kNN by a margin of about 5 points; statistical significance p<0.001). However, their performance is different in different frequency ranges: for lower frequencies kNN is more accurate (e.g., for L1 distance, p<0.001). For higher frequencies, the category-based method improves on kNN (L1, p<0.001). The centroid-based method exhibited performance, inferior to both those of kNN and the category-based method.

Tree descending algorithm. In experiments with the algorithm, candidate classes were represented in terms of the category-based method, 3 levels of hyponyms, which proved to be the best generalized representation of a class in previous experiments. Table 5 specifies the results of its evaluation.

Table 5. Tree descending algorithm.

	0-40	40-500	>500	Overall
JC	.00726 /0	.01213 /.00512	.02312 /.0101	.014904 /.005291
L1	.08221 /.03214	.05697 /.02051	.21305 /.11111	.128844 /.060846
SD	.08712 /.03214	.07739 /.03589	.16731 /.06734	.011796 /.047619

Its performance turns out to be much worse than that of the standard methods. Both direct+near and direct hits scores are surprisingly low, for 0-40 and 40-500 much lower than chance. This can be explained by the fact that some of top concepts in the tree are represented by much less distributional data than other ones. For example, there are less than 10 words that lexicalize the top concepts MASS_CONCEPT and MATHEMATICAL_CONCEPT and all of their hyponyms (compare to more than 150 words lexicalizing THING and its hyponyms up to 3 levels below it). As a result, at the very beginning of the search down the tree, a very large portion of test words was found to be similar to such concepts.

Tree ascending algorithm. The experiments were conducted with the same number of nearest neighbors as with kNN. Table 6 describes the results of evaluation of the best version (formula 3, *k*=15).

Table 6. Tree ascending algorithm, total of votes according to (3), *k*=15.

	0-40	40-500	>500	Overall
JC	.32112 /.075	.33553 /.0923	.40968 /.08754	.36643 /.08597
L1	.33369 /.07142	.34504 /.0923	.42627 /.09764	.38005 /.08862
SD	.31809 /.06785	.32489 /.05128	.45529 /.11111	.38048 /.08201

There is no statistically significant improvement on kNN overall, or in any of the frequency ranges. The algorithm favored more upper concepts and thus produced about twice as few direct hits than kNN. At the same time, its direct+near hits score was on par with that of kNN! This algorithm thus produced much more near hits than kNN, what can be interpreted as its better ability to choose a superconcept of the correct class. Based on this observation, we combined the best version of the tree ascending algorithm with kNN in one algorithm in the following manner. First the former was used to determine a superconcept of the class for the new word and thus to narrow down the search space. Then the kNN method was applied to pick a likely class from the hyponyms of the concept determined by the tree ascending method. Table 7 specifies the results of evaluation of the proposed algorithm.

Table 7. Tree ascending algorithm combined with kNN, *k*=30.

	0-40	40-500	>500	Overall
JC	**.34444** /.16428	**.35858** /.14358	**.41260** /.10774	**.38215** /.14021
L1	**.35147** /.16428	.36545 /.15384	**.41086** /.11784	**.38584** /.14682
SD	**.32613** /.13571	**.36485** /.1641	**.45732** /.16498	**.39456** /.1574

The combined algorithm demonstrated improvement both on kNN and the tree ascending method of 1 to 3 points in every frequency range and overall for direct+near hits (except for the 40-500 range, L1). The improvement was statistically significant only for L1, “>500” (p=0.05) and for L1, overall (p=0.011). For other similarity measures and frequency ranges it was insignificant (e.g., for JC, overall, p=0.374; for SD, overall, p=0.441). The algorithm did not improve on kNN in terms of direct hits. The hits scores set in bold in Table 7 are those which are higher than those for kNN in corresponding frequency ranges and similarity measures.

7. Discussion

In this paper we have examined different possibilities to take advantage of the taxonomic organization of a thesaurus to improve the accuracy of classifying new words into its classes. The study demonstrated that taxonomic similarity between nearest neighbors, in addition to their distributional similarity to the new word, may be a useful evidence on which classification decision can be based. We have proposed a “tree ascending” classification algorithm which extends the kNN method by making use of the taxonomic similarity between nearest neighbors. This algorithm was found to have a very good ability to choose a superconcept of the correct class for a new word. On the basis of this finding, another algorithm was developed that combines the tree ascending algorithm and kNN in order to optimize the search for the correct class. Although only limited statistical significance of its improvement on kNN was found, the results of the study indicate that this algorithm is a promising possibility to incorporate the structure of a thesaurus into the decision as to the class of the new word. We conjecture that the tree ascending algorithm leaves a lot of room for improvements and combinations with other algorithms like kNN.

The tree descending algorithm, a technique widely used for text categorization, proved to be much less efficient than standard classifiers when applied to the task of augmenting a domain-specific thesaurus. Its poor performance is due to the fact that in such a thesaurus there are great differences between top concepts in the amount of distributional data used to represent them, which very often misleads the top-down search.

We believe that a study of the two algorithms on the material of a larger thesaurus, where richer taxonomic information is available, can yield a further understanding of its role in the performance of the algorithms.

References

Caraballo S. A. (1999) *Automatic construction of a hypernym-labeled noun hierarchy from text*. In Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics, pp. 120-126.

Hahn U. and Schnattinger K. (1998) *Towards text knowledge engineering.* In Proc. of AAAI/IAAI, pp. 524-531.

Hearst M. and Schuetze H. (1993) *Customizing a lexicon to better suit a computational task.* In Proc. of the SIGLEX Workshop on Acquisition of Lexical Knowledge from Text, Columbus Ohio, pp. 55--69.

Grefenstette G. (1993) *Evaluation techniques for automatic semantic extraction: comparing syntactic and window based approaches*. In Proc. of the SIGLEX Workshop on Acquisition of Lexical Knowledge from Text, Columbus Ohio.

Lin D. (1998) *Automatic retrieval and clustering of similar words*. In Proc. of the COLING-ACL’98, pp. 768-773.

Lee L. (1999) *Measures of distributional similarity*. In Proc. of the 37th Annual Meeting of the Association for Computational Linguistics, pp. 25-32.

Levy J., Bullinaria J., and Patel M. (1998) *Explorations in the derivation of word co-occurrence statistics*. South Pacific Journal of Psychology, 10/1, pp. 99-111.

Maedche A. and Staab S. (2000) *Discovering conceptual relations from text*. In Proc. of ECAI-2000, IOS Press, pp. 321-324.

Pereira F., Tishby N., and Lee L. (1993) *Distributional clustering of English words.* In Proc. of the 31st Annual Meeting of the ACL, pp. 183-190.

Resnik P. (1992) *Wordnet and distributional analysis: A class-based approach to lexical discovery*. AAAI Workshop on Statistically-based Natural Language Processing Techniques.

Tokunaga T., Fujii A., Iwayama M., Sakurai N., and Tanaka H. (1997) *Extending a thesaurus by classifying words.* In Proc. of the ACL-EACL Workshop on Automatic Information Extraction and Building of Lexical Semantic Resources, pp. 16-21.