Extraction of Hypernymy Information from Text*

Erik Tjong Kim Sang, Katja Hofmann and Maarten de Rijke

Abstract We present the results of three different studies in extracting hypernymy information from text. In the first, we compare a method based on a single extraction pattern applied to the web with a set of patterns applied to a big corpus. In the second study, we examine how relation extraction can be performed reliably from a text without having access to a word sense tagger. And in a third experiment, we check what the effect of elaborate syntactic information is on the extraction process. We find that both using more data and the removal of ambiguities from the training data are beneficial to the extraction process. But to our surprise we were unable to find a positive effect of additional syntactic information.

1 Introduction

Lexical taxonomies, such as WordNet, are important resources underlying natural language processing techniques such as machine translation and word-sense disambiguation. However, manual creation and maintenance of such taxonomies are tedious and time-consuming tasks. This has led to a great deal of interest in automatic methods for retrieving taxonomic relations. Efforts for both manual development of taxonomies and automatic acquisition methods have largely focused on English-language resources. Although WordNets exist for many languages, these are usually

Erik Tjong Kim Sang
Alfa-informatica, University of Groningen, e-mail: erikt@xs4all.nl

Katja Hofmann
ISLA, University of Amsterdam, e-mail: k.hofmann@uva.nl

Maarten de Rijke
ISLA, University of Amsterdam, e-mail: derijke@uva.nl

* Parts of this paper have been published as [6, 13, 14].

much smaller than Princeton WordNet (PWN) [3], the major semantic resource for English.

For English, an excellent method has been developed for automatically extending lexical taxonomies. Snow et al. [12] show that it is possible to predict new and precise hypernym-hyponym relations from a parsed corpus. In this study, we apply their approach to data in another language (Dutch) and compare it with alternative ways for deriving hypernym-hyponym relations. We examine and answer the following three research questions:

1. How does the approach of Snow et al. perform in comparison with single-pattern extraction methods applied to data resources of various size?
2. Snow et al. relied on sense-tagged data which is presently unavailable for Dutch. What is the effect of this omission and how can we deal with it?
3. Snow et al. relied on the availability of a full parser to preprocess their data, a tool which might be missing, take a lot of time to run or produce noisy results for other languages. What results can we obtain for the relation extraction task with available cheap and reliable shallow parser preprocessing?

After this introductory section, we will describe the task, the data and the evaluation metrics in the second section. The next three sections present the findings concerning the three research questions presented above, comparison with single-pattern extraction (section 3), dealing with a missing word sense tagger (section 4) and preprocessing with a shallow parser (section 5). We conclude in section 6.

2 Task and Approach

We examine techniques for automatically extending lexical resources such as WordNets. In this section we describe which lexical relation we focus on, explain the necessary data preprocessing steps, describe the information we are looking for and introduce our evaluation approach.

2.1 Task

We will focus on extracting a particular lexical relation from text: hypernymy. One term is a hypernym of another if its meaning both covers the meaning of the second term and is broader. For example, *furniture* is a hypernym of *table*. The opposite term for hypernym is hyponym. So *table* is a hyponym of *furniture*. Hypernymy is a transitive relation. If term A is a hypernym of term B while term B is a hypernym of term C then term A is also a hypernym of term C.

In Princeton WordNet [3], hypernym relations are defined between senses of words (synsets). The Dutch WordNet (DWN), which is a part of EuroWordNet [20], contains 659,284 of such hypernym noun pairs of which 100,268 are defined hypernymy links and 559,016 are inherited by transitivity. More importantly, the resource contains hypernym information for 45,979 different nouns. A test with a recent Dutch newspaper text revealed that the Dutch WordNet only covered about two-thirds of the noun lemmas in the newspaper (among the missing words were *e-mail*, *euro* and *provider*). Proper names, like names for persons, organizations and locations, pose an even larger problem: DWN only contains 1608 words that start with a capital letter. Improving the coverage of DWN and similar resources is an important task. However, performing such a task manually is expensive, so we would like to know to what extent this task can be performed automatically.

2.2 Natural language processing

Snow et al. [12] processed their text data with natural language processing tools like a full parser. However, these tools might not be available for other languages, might be expensive and time-consuming to run or might produce unreliable output. In this study we will compare different preprocessing techniques for texts and evaluate their effects on subsequent relation extraction performance.

Our initial experiments will use as little preprocessing as possible. However, completely skipping the preprocessing step is not feasible. In the first study we apply the following preprocessing steps to the source texts:

- Tokenizing: separating punctuation marks from words and identifying sentence boundaries
- Part-of-speech tagging: assigning word classes to tokens
- Lemmatizing: assigning lemmas to tokens

All three methods are cheap in processing time. Tokenizing and lemmatizing are useful because they map related different tokens to a single token. For example, tokenizing would map *finishing.* to *finishing* and . while lemmatizing would map *finishing* to *finish*. This allows extraction patterns to be more general: we need only one pattern for *finish* rather than two extra patterns involving *finishing* and *finishing.*

For the patterns, it is very useful to be able to make a distinction between words of a different syntactic class, for example the noun and verb form of *finish*. Syntactic classes are obtained by the part-of-speech tagging step. It allows us to create extraction patterns which focus only on the noun occurrences of the target words, ignoring possible verb and adjective forms.

In our first study on the effects of pattern numbers and training data size, we will only employ these three preprocessing techniques. In our second and third study we will use full parsing. Apart from the three preprocessing steps mentioned earlier, this involves a fourth step in which the syntactic relation between pairs of words are computed. Related words might be neighbors in a sentence but they might also be words separated by several other words. This makes the full parsing step expensive in terms of computing power and computing time. We have not performed this step ourselves but have relied on a text corpus that was processed by the Dutch parser Alpino and was produced by the University of Groningen [19].

In the first study, we will also retrieve data with web queries which require plural forms of nouns. We obtain these forms from the plural list from CELEX [1] (64,040 nouns). Words that are not present in the database, receive a plural form which is determined by a machine learner trained on the database. It has the seven final characters of the words as features and can predict 152 different plural forms. Its leave-one-out accuracy on the training set is 89%.

2.3 Collecting evidence

We have employed five strategies for finding evidence for hypernymy relations between pairs of words:

1. Assume that the longest known character suffix of the hyponym is a hypernym. This morphological approach maps *blackbird* to *bird* [11]. We require the suffix to be a known word and the prefix to contain at least three characters.
2. Search a text for fixed text patterns like *A such as B and C* and consider these as evidence for relations between the word pairs *A* and *B*, and *A* and *C* [5].
3. Search a text for fixed text patterns like *A, B and C* and consider these as evidence *A*, *B* and *C* having the same hypernym [2].
4. Use a combination of the results of strategies 2 and 3
5. Starting from a list of seed pairs of related words like *A* - *B* search in the text for phrases between these words, for example *A xxx yyy B* and jointly use all of these phrases as patterns (like in 2) for finding other candidate pairs in the same context (*C xxx yyy D*) while assigning weights to the patterns based on how well they perform on the seed list [12].

The format of the text patterns that we are looking for depends on the method of text processing used. We distinguish two types of patterns: dependency patterns, which are used in combination with preprocessing by full parsing, and lexical patterns, which are used in combination with shallow preprocessing.

Dependency patterns use the dependency relations between words that are generated by the Dutch Alpino parse. For example, the parser would produce the following relations between word pairs for the sentence *Large cities in northern England such as Liverpool are beyond revival.*:

large:JJ:MOD:NN:*city*
city:NN:SUBJ:VBD:*be*
in:IN:MOD:NN:*city*
north:JJ:MOD:NNP:*England England*:NNP:OBJ1:IN:*in such*:DT:MOD:IN:*as*
as:IN:MOD:NN:*city*
Liverpool:NNP:OBJ1:IN:*as*
be:VB:–:–:–
beyond:IN:MOD:VB:*be*
revival:NN::OBJ1:IN:*beyond*

This analysis indicates that the word *large* is an adjective (JJ) which is a modifier (MOD) of head word *city* that is a noun (NN). Punctuation has been separated from words and the relations contain lemmas rather than words. Based on this syntactic analysis, we can define dependency patterns like Snow et al. [12]. Patterns are dependency paths between two nouns in the sentence with at most three intermediate nodes. Additional satellite nodes can be present next to the two nouns. Here is one of the patterns that can be derived for the two noun phrases *large cities* and *northern England* in the example sentence:

NP_1:NN:SUBJ:VBD:
in:IN:MOD:NN:NP_1
NP_2:NNP:OBJ1:IN:*in*

The pattern defines a path from the head lemma *city* via *in*, to *England*. Note that lexical information linking outside this pattern (*be* at the end of the first line) has been removed and that lexical information from the target noun phrases has been replaced by the name of the noun phrase (NP_1 at the first and second line). For each dependency pattern, we build six variants: four with additional information from the two noun phrases and two more with head information of one of the two target NPs. The pattern variants are similar to the lexical patterns variants will be presented next.

The lexical patterns only use material generated by the tokenizing, part-of-speech tagging and lemmatizing preprocessing steps. These would convert the example sentence to a sequence of lemmas with attached word class information: *large*/JJ *city*/NN *in*/IN *north*/JJ *England*/NNP *such*/DT *as*/IN *Liverpool*/NNP *be*/VB *beyond*/IN *revival*/NN *.*/. Again, *large* is an adjective (JJ) and *city* a noun (NN) but unlike with preprocessing by full parsing, the relation between the two words is undefined.

Lexical patterns contain two target phrases, both noun phrases. A maximum of three tokens can separate the two words. Additionally, the pattern may contain up to two optional extra tokens (a non-head token of the first noun phrase and/or one of the second noun phrase). The lexical preprocessing method uses two basic regular expressions for identifying noun phrases: *Determiner? Adjective* Noun+* and *ProperNoun+*. It assumes that the final token of the matched phrase is the head. Here is one set of four patterns which can be derived from the example sentence, all related to the phrase *large cities in northern England*:

1. *NP in NP*
2. *large NP in NP*
3. *NP in north NP*
4. *large NP in north NP*

The patterns contain lemmas rather than the words of the sentence in order to allow for general patterns. For the same reason, the noun phrases have been replaced by the token NP. Each of the four patterns will be used as evidence for a possible hypernymy relation between the two noun phrase heads *city* and *England*. As a novel extension to the work of Snow et al., we included two additional variants of each pattern in which either the first NP or the second NP was replaced by its head:

5. *city in NP*
6. *NP in England*

Among others, this enabled us to identify patterns containing appositions: *president NP*.

When applying extraction strategies 2 and 3, we simply select the candidate hypernym most frequently occurring in a pattern with the source hyponym. For strategy 4, we combine the two frequencies in a hypernym evidence score $s(h,w)$ for each candidate hypernym h for word w. This is the sum of the normalized evidence for the hypernymy relation between h and w, and the evidence for sibling relations between w and known hyponyms c of h:

$$s(h,w) = \frac{f_{hw}}{\sum_x f_{xw}} + \sum_c \frac{g_{cw}}{\sum_y g_{yw}}$$

where f_{hw} is the frequency of patterns that predict that h is a hypernym of w (strategy 2), g_{cw} is the frequency of patterns that predict that c is a sibling of w (strategy 3), and x and y are arbitrary words from the wordnet being used. For each word w, we select the candidate hypernym h with the largest score $s(h,w)$. We have only included evidence for hypernyms and siblings in this score. We have experimented with different scoring schemes, for example by including evidence from hypernyms of hypernyms (grandparents) and remote siblings (cousins), but found this basic scoring scheme to perform best.

2.4 Evaluation

We use the Dutch part of EuroWordNet (DWN) [20] for evaluation of our hypernym extraction methods. Hypernym-hyponym pairs that are present in the lexicon are assumed to be correct. In order for the evaluation to be complete, we also need negative examples, pairs of words that are not related by hypernymy. However, when DWN does not mention two words as being related by hypernymy, this could have several reasons. The words themselves might be missing from the resource or their relation might not have been included.

In order to have negative examples for the evaluation, we make the same assumption as Snow et al. [12]: the hypernymy relation in DWN is complete for the words that it contains. This means that when two words are present in the lexicon without the target relation being specified between them, then we assume that they are unrelated. The presence of positive and negative relations allows for an automatic evaluation in which precision, recall and F values are computed.

We do not require our extraction method to find the hypernym parent of a target word in DWN. Instead, we are satisfied with the extraction approach returning any hypernym ancestor. We use two methods in order to rule out identification methods that simply return the top node of the hierarchy for all words. In study 1, we also measure the distance between the assigned hypernym and the target word. The ideal distance is one that would occur if the ancestor is a parent. A grandparent receives distance two and so on. And in study 3, we simply disallow the top node of the hypernymy hierarchy as a candidate hypernym.

3 Study 1: Comparing Pattern Numbers and Corpus Sizes

In this first study, we apply different hypernymy extraction methods to data sources of various sizes. First, we evaluate a method for automatically deriving corpus-specific extraction patterns from a set of examples. After this we examine a method for combining these patterns and compare the performance of the combination with the best individual patterns and the morphological approach described in section 2.4. Finally, we apply two fixed patterns to web data and compare their performance with the earlier corpus results. We conclude with an analysis of the errors made by the best system.

3.1 Extracting individual patterns

In this study, we used the Twente Nieuws Corpus, a corpus of Dutch newspaper text and subtitle text covering four years (1999–2002) and containing about 300M words. The corpus was processed by automatic tools which tokenized it, assigned part-of-speech tags and identified lemmas. Next we used the same approach as Snow et al. [12] but with lexical information rather than dependency parses: all pairs of nouns with four or fewer tokens (words or punctuation signs) between them were selected. The intermediate tokens (labeled `infix`) as well as the token before the first noun (`prefix`) and the token following the second noun (`suffix`) were stored as a pattern. For each noun pair, four patterns were identified:

- `N1 infix N2`
- `prefix N1 infix N2`
- `prefix N1 infix N2 suffix`
- `N1 infix N2 suffix`

The patterns also included information about whether the nouns were singular or plural, a feature which can be derived from the part-of-speech tags. We identified 3,283,492 unique patterns. The patterns were evaluated by registering how often they assigned correct hypernym relations correspond to noun pairs from DWN. Only 118,306 patterns had a recall that was larger than zero. The majority of these patterns (63%) had a precision of 1.0 but the recall of these patterns was very low (0.00003-0.00025). The highest registered recall value for a single pattern was 0.00897 (for *N-pl and N-pl*). The recall values are low because of the difficulty of the task: we aim at generating a valid hypernym for *all* 45,979 nouns in the Dutch WordNet. A recall value of 1.0 corresponds with single pattern predicting a correct hypernym for every noun in DWN, something which is impossible to achieve.

Precision	Recall	$F_{\beta=1}$	Dist,	Pattern
0.375	0.00137	0.00273	2.56	N-pl , vooral N-pl (*especially*)
0.300	0.00133	0.00264	2.23	N-pl , waaronder N-pl (*among which*)
0.258	0.00120	0.00238	1.55	N-pl , waaronder N-sg (*among which*)
0.250	0.00196	0.00388	2.08	N-pl of ander N-pl (*or other*)
0.244	0.00418	0.00821	1.96	N-pl zoals N-sg (*such as*)
0.220	0.00259	0.00512	2.10	N-pl zoals N-pl (*such as*)
0.213	0.00809	0.01559	1.99	N-pl en ander N-pl (*and other*)
0.205	0.00387	0.00760	2.20	N-pl , zoals N-pl (*such as*)
0.184	0.00396	0.00775	1.78	N-pl , zoals N-sg (*such as*)
0.158	0.00394	0.00768	1.68	N-sg en ander N-pl (*and other*)

Table 1 Top ten high precision patterns of the format `N1 infix N2` extracted from the text corpus which have a recall score higher than 0.00100. In the patterns, N-pl and N-sg represent a plural noun and a singular noun, respectively. It is possible to aggregate patterns by ignoring the number of the noun (N-pl + N-sg = N) in order to achieve higher recall scores at the expense of lower precision rates. The phrase in parentheses is an English translation of the main words of the pattern.

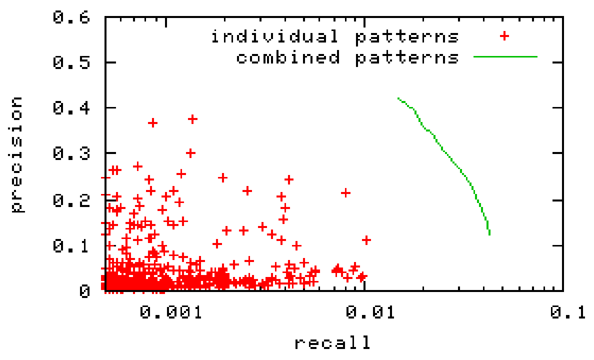

Fig. 1 Precision and recall values of the 421 hypernym-hyponym extraction patterns of the format `N1 infix N2` with the highest recall values when applied to the text corpus (+) compared with combinations of these patterns (line). Pattern combinations outperform individual patterns both with respect to precision and recall. The recall values are low because of the difficulty of the task (reproducing valid hypernyms for all nouns in the wordnet being used).

Table 1 lists ten top-precision patterns of the format `N1 infix N2` and a precision score of 0.158 or higher. Figure 1 contains an overview of the precision and recall values of all 421 patterns of that group. For comparison with other approaches, we have selected the pattern *N zoals N*, a combination of the results of four patterns of which two are listed in Table 1. This pattern obtained a precision score of 0.22 and a recall score of 0.0068 (Table 2).

Method	**Prec.**	**Recall**	**F$_{\beta=1}$**	**Dist.**
corpus: *N zoals N*	0.22	0.0068	0.013	2.01
corpus: combined	0.36	0.020	0.038	2.86
corpus: *N en N*	0.31	0.14	0.19	1.98
morphological approach	0.54	0.33	0.41	1.19

Table 2 Performance measured with the corpus approach and the morphological approach. The pattern combination perform better than the best individual pattern but both suffer from low recall figures. The conjunctive pattern and the morphological approach, predicting the longest known suffix of each word as its hypernym (section 2.4), surprisingly enough outperform both corpus approaches on most evaluation measures.

3.2 Combining corpus patterns

Snow et al. [12] showed that for the task of collecting hypernym-hyponym pairs, a combination of extraction patterns outperforms the best individual pattern. In order to obtain a combined prediction of a set of patterns, they represented word pairs by a sequence of numeric features. The value of each feature was determined by a single pattern predicting that the word pair was related according to the hypernymy relation or not. A machine learning method, Bayesian Logistic Regression was used to determine the combined prediction of feature sets for unknown word pairs based on a comparison with known word pairs which could be part of the relation or not.

We have replicated this work of Snow et al. [12] for our Dutch data. We have identified 16728 features which corresponded with hypernym-hyponym extraction patterns. All noun pairs which were associated with at least five of these patterns in the text corpus, were represented by numerical features which encoded the fact that the corresponding pattern predicted that the two were related (value 1) or not (value 0). Only nouns present in the Dutch WordNet (DWN) were considered. The class associated with each feature set could either be positive if the ordered word pair occurred in the hypernymy relation of DWN or negative if the ordered pair was not in the DWN relation. This resulted in a dataset of 528,232 different ordered pairs of which 10,653 (2.0%) were related.

The performance of the combined patterns was determined by 10-fold cross validation: the training set was divided in ten parts and the classes for each part were predicted by using the other nine parts as training data. Like Snow et al. [12] we used Bayesian Logistic Regression as learning technique [4]. We have also tested Support Vector Machines but these proved to be unable to process the data within a reasonable time.

The classifier assigned a confidence score between 0 and 1 to each pair. We computed precision and recall values for different acceptance threshold values (0.001-0.90) which resulted in the line in Figure 1. The combined patterns obtain similar precision scores as the best individual patterns but their recall scores are a lot higher. For comparison with other approaches, we have used acceptance threshold 0.5, which resulted in a precision of 0.36 and a recall of 0.020 (Table 2).

Surprisingly enough, both alternative hypernym prediction methods outperform the combination of lexical patterns (Table 2). The conjunctive pattern obtains a lower precision score than the combination but its recall is an order of magnitude larger. The morphological approach of selecting the shortest suffix that is also a valid word as the candidate hypernym (*blackbird* → *bird*), does even better: obtaining precision, recall and distance scores that are the best of all examined approaches. The morphological approach is limited in its application: it cannot find out that a *poodle* is a *dog* because the latter word is not part of the former. The relation between word pairs like these need to identified by another approach.

3.3 Web query format

Next, we will search the web for lexical patterns as evidence for lexical relations. When working with a fixed corpus on disk, an exhaustive search can be performed. For web search, however, this is not possible. Instead, we rely on acquiring interesting lexical patterns from text snippets returned for specific queries. The format of the queries has been based on three considerations.

First, a general query like * *such as* * is insufficient for obtaining much interesting information. Today, most web search engines impose a limit on the number of results returned from a query (for example 1000), which limits the opportunities for assessing the performance of such a general pattern. In order to obtain useful information, the query needs to be more specific. For the pattern * *such as* *, we have two options: adding the hypernym, which gives *hypernym such as* *, or adding the hyponym, which results in * *such as hyponym*.

Both extensions of the general pattern have their disadvantages. A pattern that includes the hypernym may fail to generate much useful information if the hypernym has many hyponyms. And patterns with hyponyms require more queries than patterns with hypernyms (at least one per child rather than one per parent). We chose to include hyponyms in the patterns. This approach models the real-world task in which someone is looking for the meaning of an unknown entity.

Our final consideration concerns the hyponyms to be used in the queries. Our focus is on evaluating the approach via comparison with an existing wordnet. Rather than flooding the search engine with queries representing every hyponym in the lexical resource, we chose to search only for a random sample of hypernyms. We observed the evaluation score to converge for approximately 1500 words and this is the number of queries we settled for.

3.4 Web extraction results

For our web extraction work, we used two fixed context patterns: one containing the word *zoals* (*such as*), a reliable and reasonably frequent hypernym pattern according to our corpus work, and another containing the word *en* (*and*), the most frequent

Method	**Prec.**	**Recall**	**F$_{\beta=1}$**	**Dist.**
web: *N zoals N*	0.23	0.089	0.13	2.06
web: *N en N*	0.39	0.31	0.35	2.04
morphological approach	0.54	0.33	0.41	1.19
web: *en* + morphology	0.48	0.45	0.46	1.64

Table 3 Performance measured in the two web experiments and a combination of the best web approach with the morphological approach. The conjunctive web pattern *N en N* rates best, because of its high frequency. All evaluation scores can be improved by supplying the best web approach with word-internal information.

pattern found in the text corpus. We chose to add randomly selected candidate hyponyms to the queries to improve the chance to retrieve interesting information.

This approach worked well. As Table 3 shows, both patterns outperformed the F-score of the combined patterns in the corpus experiments. Like in the corpus experiments, the conjunctive web pattern outperformed the *such as* web pattern with respect to precision and recall. We assume that the frequency of the two patterns plays an important role (the Google index contains about five times as many pages with the conjunctive pattern as with *zoals*).

Finally, we combined word-internal information with the conjunctive pattern approach by adding the morphological candidates to the web evidence before computing hypernym pair scores. This approach achieved the highest recall at only a slight loss in precision (Table 3). A basic combination approach by using the conjunctive pattern for searching for hyponyms for which no candidates were generated by the morphological approach, would have achieved a similar performance.

3.5 Error analysis

We have inspected the output of the conjunctive web extraction with word-internal information. For this purpose we have selected the ten most frequent hypernym pairs (top group, see Table 4), the ten least frequent (bottom group) and the ten pairs exactly between these two groups (center group): 40% of the pairs were correct, 47% incorrect and 13% were plausible but contained relations that were not present in the reference wordnet. In the center group all errors were caused by the morphological approach while all other errors in the top group and in the bottom group originated from the web extraction method.

3.6 Discussion

The contributions of our first study are two-fold. First, we showed that the large quantity of available web data allows basic patterns to perform better on hypernym extraction than an advanced combination of extraction patterns applied to a large

+/-	score	hyponym	hypernym
-	912	buffel	predator
+	762	trui	kledingstuk
?	715	motorfiets	motorrijtuig
+	697	kruidnagel	specerij
-	680	concours	samenzijn
+	676	koopwoning	woongelegenheid
+	672	inspecteur	opziener
?	660	roller	werktuig
?	654	rente	verdiensten
?	650	cluster	afd.

Table 4 Example output of the conjunctive web system with word-internal information. Of the ten most frequent pairs, four are correct (+). Four others are plausible but are missing (?) in the wordnet used for evaluation.

corpus. Second, we demonstrate that the performance of the web extraction method can be improved by combining its results with those of a corpus-independent morphological approach.

While the web results are of reasonable quality, some concern can be expressed about the quality of the corpus results. At best, we obtained an F-value of 0.038, a lot lower than than the 0.348 reported for English in [12]. There are two reasons for this difference. First, the evaluation methods are different: we aim at generating hypernyms for all words in the wordnet that we use while Snow et al. only look for hypernyms for words in the wordnet *that are present in their corpus*. Second, in their extraction work Snow et al. also use a sense-tagged corpus, a resource which is unavailable for Dutch at present. In the next section, we will examine this difference in more detail.

4 Study 2: Examining the Influence of Ambiguity

In our second study, we apply the relation extraction method of Snow et al. [12] to a non-English language (Dutch) for which only a basic wordnet and a parser are available. We find that, without an additional sense-tagged corpus, this approach is highly susceptible to noise due to word sense ambiguity. We propose and evaluate two methods to address this problem.

4.1 Approach

Our approach closely follows the approach described in Snow et al. [12]. From a parsed newspaper corpus we select all sentences containing nouns that can be found in the Dutch part of EuroWordNet (DWN) [20]. We extract ordered noun pairs from each sentence and label the noun pairs as *Known Hypernym* if they are related according to the transitive closure of the hypernym relation in DWN. All other noun pairs occurring in DWN are labeled *Known Non-Hypernym*, conform the completeness assumption described in section 2.4. Next, we collect all dependency paths between noun pairs and use them as features in a machine learning experiment for predicting Known Hypernym pairs. We ignore paths that occur with fewer than five unique noun pairs as well as noun pairs which appear with fewer than five different dependency paths.

As we saw in the previous section, our initial implementation performed much worse than the results reported in [12] (F-score of 0.11 vs. 0.348). A major difference between the two systems, was that Snow et al. [12] uses frequency counts for word-senses from a sense-tagged corpus to reduce ambiguity in their training data. As no such resource is available for Dutch, we explored alternative ways of reducing ambiguity in our training data. We propose two methods for reducing ambiguity and compare performance at different levels of ambiguity. First, we assume that DWN assigns the first sense (i.e., lowest sense number) to the most frequent sense of a polysemous word. Following this assumption, we filter training instances labeled as *Known Hypernym* to only include noun pairs where a hypernym relation exists for the first senses of both nouns. Our second method circumvents the problem of ambiguity by including only those training instances where both nouns are monosemous (i.e., have only one sense).

4.2 Experiments and results

Our experiments are based on a part of the Twente News Corpus: (TwNC-02), consisting of over 23 million sentences from Dutch newspaper articles. The corpus was parsed with Alpino [15]. We generate dependency patterns from the parses by moving the category of the *head* child of each node into the parent node. We add satellite links to the words preceding and following the pattern in the sentence. Thus, for each word pair we extract up to four patterns.

Using all DWN nouns, the ratio of negative to positive examples in the corpus is approximately 38:1 compared to 50:1 in [12]. We attribute our relatively high number of positive instances to a large number of false positives due to word-sense ambiguity. For the reduced-ambiguity data sets the percentage of negative instances is much higher. Like Snow et al. [12], we restrict the ratio of negative to positive items in these data sets to 50:1 by randomly removing negative items.

We train a Logistic Regression[2] classifier to predict hypernym relations between word pairs using binary features. Each feature represents the occurrence of a dependency path for a given noun pair. We work with a single data set which is evaluated using 10-fold cross validation.

We performed five experiments. In (a) we included all 45,981 nouns from DWN. (b) used only positive instances where the hypernym relation applied to the first

[2] http://stat.rutgers.edu/∼madigan/BBR/

	# pairs	# pos	# neg	Prec.	Rec.	$F_{\beta=1}$
a	253,068	5,108	247,960	0.067	0.078	0.072
b	250,786	5,041	245,745	0.148	0.154	0.151
c	251,127	5,115	246,012	**0.215**	**0.302**	**0.251**
d	1,718,369	43,267	1,675,102	0.085	0.160	0.111
e	1,093,150	21,035	1,072,115	0.136	0.209	0.165

Table 5 Results from training with: (a+d) all DWN noun data; (b+e) only the first sense of each noun; and (c) monosemous nouns only.

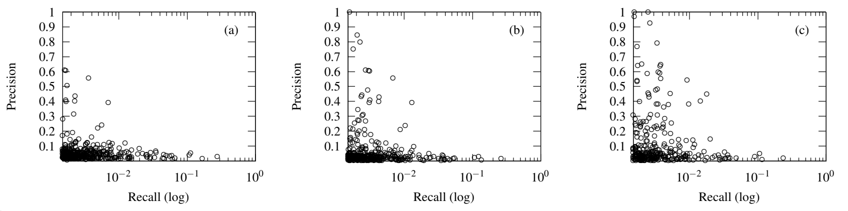

Fig. 2 Predictive quality of individual dependency patterns extracted using (a) all nouns in DWN; (b) first noun senses only; and (c) monosemous words only. Each data point represents precision and recall of one dependency pattern.

sense of each noun. (c) was restricted to the 40,069 monosemous DWN nouns. In order to cancel the benefits of larger training sets, the size of the first three data sets was limited to the size of the smallest, (c), through random selection of training instances. Experiments (d) and (e) were performed with the complete data sets of (a) and (b) respectively.

Table 5 shows precision, recall and $F_{\beta=1}$ of the machine learner for each data set. Results obtained with the monosemous data set are significantly ($p \ll 0.001$, estimated with bootstrap resampling) better than those obtained with all data, even for data sets which are considerably larger. Figure 2 shows that the largest number of high-precision patterns was identified using the monosemous data set.

Our best F-score (0.251) is lower than the best score reported in [12] (0.348). We suspect that the prime reason for this is the size difference between the bootstrap lexical resources: PWN (114,648 nouns) and DWN (45,981). This allows the English work to derive many more positive sense-restricted examples (14,387 in comparison with the 5,115 for Dutch) even though we have access to a larger parsed corpus (23 million vs. 6 million sentences).

4.3 Discussion

The contributions of our second study are two-fold. First, we confirm the results of [12] and show that the method of automatically extracting dependency patterns for hypernym acquisition can be transfered to languages other than English. Second, we show that the method is sensitive to word sense ambiguity but that this problem can be addressed by removing polysemous nouns from the training data.

The results obtained in this study improve on earlier work on extracting Dutch hypernym relations. IJzereef [7] reports a precision score of 0.198 for extracting hypernym relations with a small set of fixed patterns (Table 5.3, recall figures have not been included). Van de Plas and Bouma [16] use a more complex evaluation score which cannot easily be compared with our evaluation approach.

A large amount of followup work could be done, ranging from further evaluation to applications of the method in larger contexts. Our next step will be manual evaluation of the method, as a comparison to DWN is not fully sufficient for assessing its actual performance. Further evaluation will also allow us to assess sensitivity of the approach to factors other than ambiguity, such as corpus size, differences in dependency parses, or the use of lexical patterns instead of dependency parses.

Another promising direction for future work would be to exploit features specific to the Alpino dependency parser, such as multi-word phrases. These elements usually contain named entities, which would be interesting to add to DWN as it currently contains few named entities. We are also planning to apply the method of automatic dependency pattern extraction to extending DWN and use it for deriving relations other than hypernymy.

5 Study 3: Measuring the Effect of Syntactic Processing

In our third and final study, we examine the effect of two text preprocessing approaches on the task of extracting hypernymy information. The first of the two methods is shallow linguistic processing, a robust and fast text analysis method which only uses information from words, like lemma information and part-of-speech classes. The second method is dependency parsing, which includes information about the syntactic relations between words. The task, the preprocessing methods and the evaluation setting have already been described in sections 2.1, 2.2 and 2.4, respectively. In the next section, we will show how our experiments were set up and present the results. After that we will discuss the effect of the two methods on the extraction task.

5.1 Experiments and results

We have applied the extraction techniques to two different Dutch corpora. The first is a collection of texts from the news domain. It consists of texts from five different Dutch news papers from the Twente News Corpus collection. Two versions of this corpus exist. We have worked with the version which contains the years 1997-2005 (26 million sentences and 450 million tokens). The second corpus is the Dutch Wikipedia. Here we used a version of October 2006 (5 million sentences and 58 million words).

Lexical patterns

Data source	**Targ.**	**Prec.**	**Recall**	**F$_{\beta=1}$**
AD	620	55.8%	27.9%	37.2
NRC	882	50.4%	23.8%	32.3
Parool	462	51.8%	21.9%	30.8
Trouw	607	54.1%	25.9%	35.0
Volkskrant	970	49.7%	24.1%	32.5
Newspapers	3307	43.1%	26.7%	33.0
Wikipedia	1288	63.4%	44.3%	52.1

Dependency patterns

Data source	**Targ.**	**Prec.**	**Recall**	**F$_{\beta=1}$**
AD	706	42.9%	30.2%	35.4
NRC	1224	26.2%	25.3%	25.7
Parool	584	31.2%	23.8%	27.0
Trouw	760	35.3%	29.0%	31.8
Volkskrant	1204	29.2%	25.5%	27.2
Newspapers	3806	20.7%	29.1%	24.2
Wikipedia	1580	61.9%	47.0%	53.4

Table 6 Hypernym extraction scores for the five newspapers in the Twente News Corpus (AD, NRC, Parool, Trouw and Volkskrant) and for the Dutch Wikipedia. The Targets column shows the number of unique positive word pairs in each data set. The Dutch Wikipedia contains about as much data as one of the newspaper sections.

Syntactic preprocessing of the material was done with the Alpino parser, the best available parser for Dutch with a labeled dependency accuracy of 89% [18]. Rather than performing the parsing task ourselves, we have relied on an available parsed treebank which included the text corpora that we wanted to use [19].

The parser also performs part-of-speech tagging and lemmatization, tasks that are useful for the lexical preprocessing methods. However, taking future real-time applications in mind, we did not want the lexical processing to be dependent on the parser. Therefore we have developed an in-house part-of-speech tagger and lemmatizer based on the material created in the Corpus Spoken Dutch project [17]. The tagger achieved an accuracy of 96% on test data from the same project while the lemmatizer achieved 98%.

We used the Dutch part of EuroWordNet [20] as the gold standard lexical resource, both for training and testing. In the lexicon, many nouns have different senses. This can cause problems for the pattern extraction process. For example, if a noun N_1 with sense X is related to another noun N_2 then the appearance of N_1 with sense Y with N_2 in the text may be completely accidental and say nothing about the relation between the two words. In that case it would be wrong to regard the context of the two words as an interesting extraction pattern. We avoided this problem by using the approach described in the previous section, removing all nouns with multiple senses from the data set and use only the monosemous words for finding good extraction patterns. This restriction is only imposed in the training phase. We consider both monosemous words and polysemous words in the evaluation process.

We imposed two additional restrictions on the lexical resource. First, we removed the top noun of the hypernymy hierarchy (*iets*) from the list of valid hypernyms. This word is a valid hypernym of any other noun. It is not an interesting suggestion for the extraction procedure to put forward. Second, we restricted the extraction procedure to propose only known hypernyms as candidate hypernyms. Nouns that appeared in the lexical resources only as hyponyms (leaf nodes of the hypernymy tree) were never proposed as candidate hypernyms. This made sense for our evaluation procedure which is only aimed at finding known hypernym-hyponym pairs.

We performed two hypernym extraction experiments, one which used lexical extraction patterns and one which used dependency patterns.[3] The results from the experiments can be found in Table 6. The newspaper F-scores obtained with lexical patterns are similar to those reported for English [12, 32.0] but the dependency patterns perform worse. Both approaches perform well on Wikipedia data, most likely because of the more repeated sentence structures and the presence of many definition sentences. For newspaper data, lexical patterns outperform dependency patterns both for precision and $F_{\beta=1}$. For Wikipedia data the differences are smaller and in fact the dependency patterns obtain the best F-score. For all data sets, the dependency patterns suggest more related pairs than the lexical patterns (column Targets). The differences between the two pattern types are significant ($p < 0.05$) for all evaluation measures for Newspapers and for positive targets and recall for Wikipedia.

5.2 Result analysis

In this section, we take a closer look at the results described in the previous section. We start with looking for an explanation for the differences between the scores obtained with lexical patterns and dependency patterns. First we examine the results for Wikipedia data and then the results for newspaper data. Finally, we perform an error analysis to find out the strengths and weaknesses of each of the two methods.

The most important difference between the two pattern types for Wikipedia data is the number of positive targets (Table 6). Dependency patterns find 23% more related pairs in the Wikipedia data than lexical patterns (1580 vs. 1288). This effect can also be simulated by changing the size of the corpus. If we restrict the data set of the dependency patterns to 70% of its current size then the patterns retrieve a similar number of positive targets as the lexical patterns, 1289, with comparable precision, recall and $F_{\beta=1}$ scores (62.5%, 46.6% and 53.4). So we expect that the effect of applying the dependency patterns is the same as applying the lexical patterns to 43% more data.

Performance-wise there seems to be only a small difference between the two preprocessing methods when applied to the Wikipedia data set. However, when we examine the scores obtained on the newspaper data (Table 6) then we find larger differences. Dependency patterns find more positive targets and obtaining a larger recall score but their precision score is disappointing. However, when we examine the precision-recall plots of the two methods (Figure 3, obtained by varying the acceptance threshold of the machine learner), they are almost indistinguishable. The performance line for lexical patterns extends further to the left than the one of the dependency patterns but the remainder of the two graphs overlap. The measured performances in Table 6 are different because the machine learner put the acceptance level for extracted pairs at different points of the graph: the performance

[3] The software used in these experiment has been made available at http://www.let.rug.nl/erikt/cornetto/D08.zip.

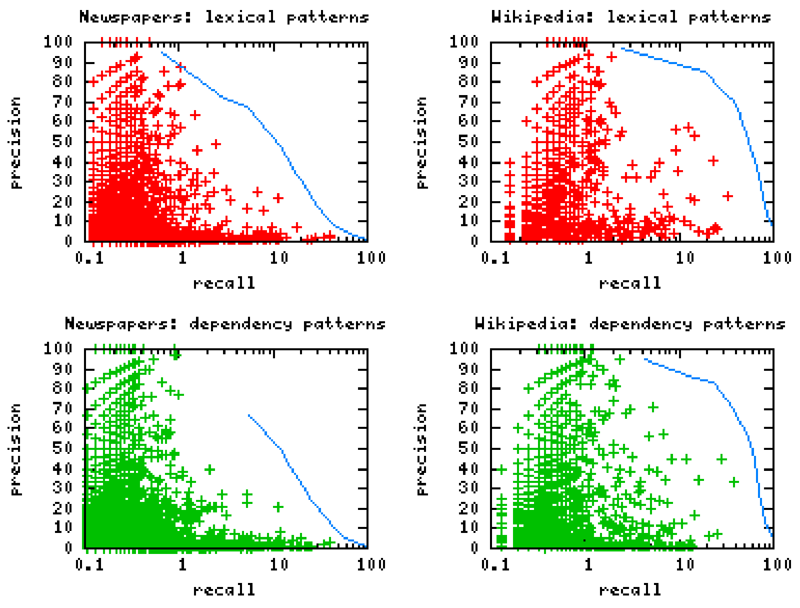

Fig. 3 Performance of individual hypernym extraction patterns applied to the combination of five newspapers and Wikipedia. Each + in the graphs represent a different extraction pattern. The precision-recall graphs for the machine learner (lines) are identical for each data source except for the extended part of the performance line for lexical patterns.

lexical patterns applied to Newspapers

Key Phrase	**Targ.**	**Prec.**	**Recall**	**$F_{\beta=1}$**
N and other N	376	22.0%	11.4%	15.0
N such as N	222	25.1%	6.7%	10.6
N like N	579	7.6%	17.5%	10.6
N , such as N	263	15.6%	8.0%	10.5
N (N	323	7.5%	9.8%	8.5

Dependency patterns applied to Newspapers

Key Phrase	**Targ.**	**Prec.**	**Recall**	**$F_{\beta=1}$**
N and other N	420	21.1%	11.0%	14.5
N be a N	451	8.2%	11.8%	9.7
N like N	205	27.3%	5.4%	9.0
N be N	766	5.7%	20.1%	8.8
N such as N	199	22.4%	5.2%	8.5

Lexical patterns applied to Wikipedia

Key Phrase	**Targ.**	**Prec.**	**Recall**	**$F_{\beta=1}$**
N be a N	294	40.8%	22.8%	29.3
N be N	418	22.9%	32.5%	26.9
a N be N	185	53.3%	14.4%	22.6
N such as N	161	57.5%	12.5%	20.5
N (N	188	21.2%	14.6%	17.3

Dependency patterns applied to Wikipedia

Key Phrase	**Targ.**	**Prec.**	**Recall**	**$F_{\beta=1}$**
N be N	609	33.6%	38.5%	35.9
N be a N	452	44.3%	28.6%	34.8
the N be N	258	34.0%	16.3%	22.1
a N be N	184	44.7%	11.6%	18.5
N N	234	16.6%	14.8%	15.6

Table 7 Best performing extraction patterns according to F-scores.

lines in both newspaper graphs contain (recall,precision) points (26.7%,43.1%) and (29.1%,20.7%).

We are unable to find major differences in the results of the two approaches. We conclude that, apart from an effect which can be simulated with some extra data,

56	—	covered by other patterns	45	—	covered by other patterns
12	48%	required full parsing	38	64%	parsing errors
6	24%	lemmatization errors	10	17%	lemmatization errors
3	12%	omitted for lack of support	7	12%	extraction pattern errors
3	12%	pos tagging errors	3	5%	omitted for lack of support
1	4%	extraction pattern error	1	2%	pos tagging error
81	100%		104	100%	

Table 8 Primary causes of recall errors made by the lexical pattern *N such as N* (top) and the best performing corresponding dependency pattern (bottom).

there is no difference between preprocessing text with shallow methods and with a full dependency parser.

Despite the lack of performance differences between the two preprocessing methods, there are still internal differences which cause one method to generate different related word pairs than the other. Next, we will examine in detail two extraction patterns and specify their distinct effects on the output results. We hope that by carefully examining their output we can learn about the strengths and weaknesses of the two approaches.

We take a closer look at extraction pattern *N such as N* for Newspaper data (second best for lexical patterns and fifth best for dependency patterns, see Table 7). The lexical pattern found 222 related word pairs while the dependency pattern discovered 199. 118 of these pairs were found by both patterns which means that the lexical pattern missed 81 of the pairs while the dependency pattern missed 104.

An overview of the cause of the recall errors can be found in Table 8. The two extraction patterns do not overlap completely. The dependency parser ignored punctuation signs and therefore the dependency pattern covers both phrases with and without punctuation. However, these phrase variants result in different lexical patterns. This is the cause for 56 hypernyms being missed by the lexical pattern. Meanwhile there is a difference between a dependency pattern without the conjunction *and* and one with the conjunction, while there is a unified lexical pattern processing both phrases with and without conjunctions. This caused the dependency pattern to miss 45 hypernyms. However, all of these 'missed' hypernyms are handled by other patterns.

The main cause of the recall differences between the two extraction patterns was the parser. The dependency pattern found twelve hypernyms which the lexical pattern missed because they required an analysis which was beyond part-of-speech tagging and the basic noun phrase identifier used by the lexical preprocessor. Six hypernyms required extending a noun phrase with a prepositional phrase, five needed noun phrase extension with a relative clause and one involved appositions. An example of such a phrase is *illnesses caused by vitamin deficits, like scurvy and beriberi*.

However, the syntactic information that was available to the dependency pattern did also have a negative effect on its recall: 38 of the hypernyms detected by the lexical pattern were missed by the dependency pattern because there was a parsing error in the relevant phrase. In more than half of the cases, this involved attaching the phrase starting with *such as* at an incorrect position. We found that a phrase like N_1 *such as* N_2 , N_3 *and* N_4 could have been split at any position. We even found some cases of prepositional phrases and relative clauses incorrectly being moved from other positions in the sentence into the target phrase.

Other recall error causes appear less frequently. The two preprocessing methods used different lemmatization algorithms which also made different errors. The effects of this were visible in the errors made by the two patterns. Some hypernyms that were found by both patterns but were not present in both results because of insufficient support from other patterns (candidate hypernyms should be supported by at least five different patterns). The effect of errors in part-of-speech tags was small. Our data analysis also revealed some inconsistencies in the extraction patterns which should be examined.

5.3 Discussion

We have evaluated the effects of two different preprocessing methods for a natural language processing task: automatically identifying hypernymy information. The first method used lexical patterns and relied on shallow processing techniques like part-of-speech tagging and lemmatization. The second method used dependency patterns which relied on additional information obtained from dependency parsing.

In earlier work, McCarthy et al. [10] found that for word sense disambiguation using thesauri generated from dependency relations perform only slightly better than thesauri generated from proximity-based relations. Jijkoun et al. [8] showed that information obtained from dependency patterns significantly improved the performance of a question answering system. Li and Roth [9] report that preprocessing by shallow parsing allows for a more accurate post-processing of ill-formed sentences than preprocessing with full parsing.

Our study supports the findings of McCarthy et al. [10]. We found only minor differences in performances between the two preprocessing methods. The most important difference: about 20% extra positive cases that were identified by the dependency patterns applied to Wikipedia data, can be overcome by increasing the data set of the lexical patterns by half. We believe that obtaining more data may often be easier than dealing with the extra computing time required for parsing the data. For example, in the course of performing this study, we had to refrain from using a recent version of Wikipedia because parsing the data would have taken 296 *days* on a single processor machine compared with a single hour for tagging the data.

6 Concluding Remarks

We have presented three studies in automatic relation extraction. In the first study, we showed that fixed extraction patterns applied to web data were able to extract more related words than a set of derived patterns applied to corpus data. Rules based on the morphological structure of words performed even better but have a limited application. The best-performing extraction approach was a combination of web patterns and morphological patterns.

In the second study, we examined the effect of ambiguity on the extraction process. We observed a distinctive effect: the results improved when we removed positive examples related to non-prominent word senses from the training data. Results got even better when all ambiguous words were removed from the training data alltogether. These two approaches are very suitable for performing relation extraction for languages for which no word-sense tagger is available.

The third study measured the influence of including elaborate syntactic information in the extraction process. We found only a small positive effect of this information, one which could also be obtained with a combination of shallow syntactic processing and a limited amount of extra training data. This is also a good observation for future application of these extraction techniques to languages for which only few language processing tools are available.

Acknowledgements

The research leading to these results was supported by the Cornetto and DuOMAn projects carried out within the STEVIN programme which is funded by the Dutch and Flemish Governments (http://www.stevin-tst.org), by the Netherlands Organisation for Scientific Research (NWO) under project nrs 264.70.050, 612.066.512, 612.061.814, 612.061.815, 640.004.802 and by the European Union's ICT Policy Support Programme as part of the Competitiveness and Innovation Framework Programme, CIP ICT-PSP under grant agreement nr 250430.

References

[1] R.H. Baayen, R. Piepenbrock, and L. Gulikers. *The CELEX Lexical Database (Release 2) [CD-ROM].* Philadelphia, PA: Linguistic Data Consortium, University of Pennsylvania, 1995.
[2] Sharon A. Caraballo. Automatic construction of a hypernym-labeled noun hierarchy from text. In *Proceedings of ACL-99*. Maryland, USA, 1999.
[3] Christiane Fellbaum. *WordNet – An Electronic Lexical Database*. The MIT Press, 1998.
[4] Alexander Genkin, David D. Lewis, and David Madigan. *Large-Scale Bayesian Logistic Regression for Text Categorization*. Technical report, Rutgers University, New Jersey, 2004.
[5] Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of ACL-92*. Newark, Delaware, USA, 1992.

[6] Katja Hofmann and Erik Tjong Kim Sang. Automatic extension of non-english wordnets. In *Proceedings of SIGIR'07*. Amsterdam, The Netherlands, 2007. (poster).

[7] Leonie IJzereef. *Automatische extractie van hyperniemrelaties uit grote tekstcorpora*. MSc thesis, University of Groningen, 2004. (in Dutch).

[8] Valentin Jijkoun, Maarten de Rijke, and Jori Mur. Information extraction for question answering: Improving recall through syntactic patterns. In *Proceedings of Coling'04*. Geneva, Switzerland, 2004.

[9] Xin Li and Dan Roth. Exploring evidence for shallow parsing. In *Proceedings of Conference on Computational Natural Language Learning (CoNLL) 2001*, 2001.

[10] Diana McCarthy, Rob Koeling, Julie Weeds, and John Caroll. Unsupervised acquisition of predominant word senses. *Computational Linguistics*, 33(4), 2007.

[11] Marta Sabou, Chris Wroe, Carole Goble, and Gilad Mishne. Learning domain ontologies for web service descriptions: an experiment in bioinformatics. In *14th International World Wide Web Conference (WWW2005)*. Chiba, Japan, 2005.

[12] Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. Learning syntactic patterns for automatic hypernym discovery. In *NIPS 2005*. Vancouver, Canada, 2005.

[13] Erik Tjong Kim Sang. To use a treebank or not – which is better for hypernym extraction. In *Proceedings of the Seventh International Workshop on Treebanks and Linguistic Theories (TLT 7)*. Groningen, The Netherlands, 2009.

[14] Erik Tjong Kim Sang and Katja Hofmann. Automatic extraction of dutch hypernym-hyponym pairs. In *Proceedings of CLIN-2006*. Leuven, Belgium, 2007.

[15] L. van der Beek, G. Bouma, R. Malouf, and G. van Noord. The alpino dependency treebank. In *Proceedings of CLIN 2001*, Twente University, 2002.

[16] Lonneke van der Plas and Gosse Bouma. Automatic acquisition of lexicosemantic knowledge for qa. In *Proceedings of the IJCNLP Workshop on Ontologies and Lexical Resources*. Jeju Island, Korea, 2005.

[17] Frank Van Eynde. *Part of Speech Tagging en Lemmatisering van het Corpus Gesproken Nederlands*. K.U. Leuven, 2005. (in Dutch).

[18] Gertjan van Noord. At last parsing is now operational. In Piet Mertens, Cedrick Fairon, Anne Dister, and Patrick Watrin, editors, *TALN06. Verbum Ex Machina. Actes de la 13e conference sur le traitement automatique des langues naturelles*, 2006.

[19] Gertjan van Noord. Huge parsed corpora in lassy. In *Proceedings of TLT7*. LOT, Groningen, The Netherlands, 2009.

[20] Piek Vossen. *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*. Kluwer Academic Publisher, 1998.