Semantic Clustering in Dutch

Automatically inducing semantic classes from large-scale corpora

Tim Van de Cruys

CLCG, University of Groningen

Abstract

Handcrafting semantic classes is a difficult and time-consuming job, and depends on human interpretation. Possible machine learning techniques would be much faster, and do not rely on interpretation, because they stick to the data. The goal of this research is to present some machine learning techniques that make it possible to achieve an automatic clustering of Dutch words. More particularly, vector space measures are used to compute the semantic similarity of nouns according to the adjectives those nouns collocate with. Such semantic similarity measures provide a thorough basis to cluster nouns into semantic classes. Partitional clustering algorithms, that produce stand-alone clusters, as well as agglomerative clustering algorithms, that produce hierarchical trees, are investigated. For the evaluation of the clusters, evaluation frameworks will be used that compare the clusters to the hand-crafted Dutch EuroWordNet and the Interlingual Wordnet synsets. Additionally, the clustering of adjectives according to the collocating nouns has been investigated.

1 Introduction

Automatically acquiring semantics from text is a subject that has gathered a lot of attention for quite some time now. As Manning and Schütze (2000) point out, most work on acquiring semantic properties of words has focused on *semantic similarity*. Automatically acquiring a relative measure of how similar a word is to known words (...) is much easier than determining what the actual meaning is.

Most work on semantic similarity relies on the Distributional Hypothesis (Harris 1985). This hypothesis states that words that occur in similar contexts tend to be similar. Take for example the invented word *sneup*, used in a number of contexts:

- verse sneup
- gezouten sneup
- lekkere sneup
- zoete sneup
- taaie sneup

A speaker of Dutch who is not familiar with the word *sneup* can easily infer from the context that it is some kind of food. In the same way, a computer might be able to extract similar words from similar contexts, and group them into clusters. There are, however, some problems with such an automatic approach. Ambiguity is the most important problem. Take the examples:

- een oneven nummer

- een steengoed nummer

The word *nummer* does not have the same meaning in these examples. In the first sense, *nummer* is used in the sense of 'designator of quantity'. In the second sense, it is used in the sense of 'musical performance'. Accordingly, we would like the word *nummer* to end up in two different clusters, the first cluster consisting of words like *getal, cijfer* and the second cluster containing words like *liedje, song*.

While it is relatively easy for a human language user to distinguish between the two senses, this is a difficult task for a computer. Moreover, the results get blurred because the attributes of both senses (in this example *oneven* and *steengoed*) are grouped together. Although an active disambiguation approach has not been pursued in this research, some algorithms are discussed later on that might be able to resolve this kind of ambiguity, and distinguish between the different senses of a word.

2 Concepts and Methodology

2.1 Vector Space Measures

The actual semantic similarity of words is determined by means of vector space measures. The two words, for which the semantic similarity is to be calculated, are represented as vectors in a multi-dimensional space. There are two possible vector space representations: binary vector spaces and real-valued vector spaces. **Binary vectors** only have one bit of information on each dimension. For linguistic objects, the **real-valued vector space** is more appropriate, as this makes it possible to encode the frequency of the attribute.

In this research, the vector space consists of the adjectives (modifiers) of the nouns. Figure 1 gives an example of four nouns represented as vectors in *modifier space*.[1]

	rood	lekker	snel	tweedehands
appel	2	1	0	0
wijn	2	2	0	0
auto	1	0	1	2
vrachtwagen	1	0	1	1

Figure 1: A noun-by-adjective matrix

The matrix shows that the modifier *rood* collocates with all four nouns, while *lekker* only collocates with *appel* and *wijn*. On the other hand, *snel* and *tweedehands* only collocate with *auto* and *vrachtwagen*.

[1] N.B. In the actual clustering framework, the frequency of the adjectives has been logarithmically smoothed ($f(x) = 1 + ln(x) \quad for\ each\ x > 0$), in order to normalize the occurrence of many instances of one single adjective.

This example shows how it might be possible to make a judgement about the similarity of nouns according to the collocating adjectives. But to make this approach really useful, an appropriate similarity measure is needed. Such a measure is discussed below.

2.2 Similarity measure

Several similarity measures are available to calculate the similarity among various patterns. A few possibilities are *Dice coefficient*, *Jaccard coefficient* and *Overlap coefficient*. In these experiments, the *cosine measure* has been used. The cosine measure penalizes less in cases where the number of non-zero entries is very different. This is more appropriate in linguistic contexts, since the amount of data available for certain words might be different, and we do not want to qualify words as dissimilar because of this property.

For the general case of two n-dimensional vectors $\overrightarrow{x}$ and $\overrightarrow{y}$ in a real-valued space, the cosine measure can be calculated as follows:

$$cos(\overrightarrow{x}, \overrightarrow{y}) = \frac{\overrightarrow{x} \cdot \overrightarrow{y}}{\mid \overrightarrow{x} \mid\mid \overrightarrow{y} \mid} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i^2}}$$

This formula yields a number between 0 and 1, where 0 means no similarity at all and 1 means identical vectors. When applying the cosine similarity measure to the vectors in figure 1, we get:

- $cos(appel, wijn) = \frac{6}{\sqrt{40}} \cong 0.94$
- $cos(auto, vrachtwagen) = \frac{4}{\sqrt{18}} \cong 0.94$
- $cos(appel, vrachtwagen) = \frac{2}{\sqrt{15}} \cong 0.51$

These simple examples show how semantically similar words are found: semantically similar words get high cosine values due to equal contexts of collocating adjectives, while semantically dissimilar words get a lower cosine value because of differing collocating adjectives.

2.3 Clustering

There are various clustering methods. In general, a distinction can be made between:

- partitional clustering algorithms: algorithms that produce ‘stand-alone’ clusters which are not embedded in a structure;
- agglomerative (hierarchical) clustering algorithms: algorithms that assign a complete branching structure to the various clusters, up to the root node.

Both algorithms are interesting to explore in the framework of semantic clustering. Partitional clustering is interesting to check whether similar words get grouped together. Hierarchical clustering is interesting to see whether this kind of clustering is able to produce a sensical wordnet, that is comparable to handcrafted wordnets. Both approaches have been explored in this internship.

2.3.1 Partitional clustering

As a partitional algorithm, K-means has been used. The procedure of K-means is as follows:

1. Choose *k* cluster centers, which are usually *k* randomly-chosen patterns or *k* randomly defined points inside the vector space;
2. assign each pattern to the closest cluster center;
3. recompute the cluster centers using the current cluster memberships;
4. if a convergence criterion is not met, go to step 2. Otherwise, stop the algorithm. The convergence criterium might be: no (or minimal) reassignment of patterns to new cluster centers, or a minimal decrease in squared error (a measure to check whether there are still many differences in cluster assignment in each iterative step).

2.3.2 Hierarchical clustering

There are three well-known algorithms for hierarchical clustering: single-link, complete-link and group-average agglomerative clustering. These algorithms only differ in the way they characterize the similarity between a pair of clusters. In the single-link method, the distance between two clusters is the minimum of the distances between all pairs of patterns drawn from the two clusters (one pattern from the first cluster, the other from the second). In the complete-link algorithm, the distance between two clusters is the maximum of all pairwise distances between patterns in the two clusters. In group-average agglomerative clustering, the distance between two clusters is the average distance between patterns in the two clusters. In the three algorithms, two clusters are merged to form a larger cluster based on minimum distance criteria.

As the group-average agglomerative clustering algorithm has been used in this research, the procedure of this algorithm is elaborated below:

1. The algorithm starts by taking each individual pattern in the pattern set to form a cluster;
2. next, the two clusters which are most similar are grouped together. Most similar means: the two clusters with the smallest distance between the averages of the clusters;
3. step two is repeated until there is only one cluster left. When the algorithm terminates, all clusters are hierarchically connected to the root node.

2.3.3 Hard and soft clustering

An extra distinction needs to be made between hard clustering and soft clustering algorithms. In hard clustering algorithms, each element is assigned to exactly one cluster. In soft clustering algorithms, an element may be assigned to several clusters. Usually, soft clustering algorithms yield a probability distribution for each pattern, in which some patterns are more likely to belong to certain clusters than to others. This might seem a tempting approach for natural language processing, because ambiguity requires some words to be assigned to several clusters. However, soft clustering, as it is generally understood, is not the most appropriate approach to cope with disambiguous words. Since all the attributes of ambiguous words are taken into account (attributes that belong to different senses of the word), the vector that is constructed cannot represent both senses of the word, but it will present some kind of average, in which the most dominant sense will have the upper hand. Therefore, only hard clustering approaches have been pursued (equally running the risk of wrong cluster assignment with ambiguous words, and missing out on less frequent senses of an ambiguous word).

There is, however, another soft clustering approach. In this approach, an ambiguous word is first assigned to a (dominant) sense of the word (found with all the attributes of the word). Once assigned to a certain cluster, the attributes that belong to this cluster are removed from the word vector, so that other, less common senses of the word might be revealed. Such algorithms are called *disjunctive clustering models*. It might be interesting to develop such an algorithm for Dutch. The algorithm discussed by Pantel and Lin (2002) would be a good algorithm to start from. This algorithm indeed tries to find less common word senses by stripping the values of more common senses off the feature vector. However, finding less common senses of a word most likely requires much more data.

2.4 Experimental Design

All noun-adjective collocations have been extracted from the Twente News Corpus (TwNC). The corpus was tagged with Mbt (Daelemans, Zavrel et al. 1996), a memory-based tagger, and lemmatized with Mblem (van den Bosch and Daelemans 1999), a memory-based lemmatizer.

Various parameters have been used for clustering, but a combination of the 5,000 most frequent nouns (adjectives) together with the 20,000 most frequent adjectives (nouns) seems to be functioning best (clustering more nouns yields much worse cluster quality, clustering the same nouns with more adjectives does not yield any significant improvement).

When using a corpus of this size, the quality of the clusters is significantly better.

3 Results

3.1 Partitional clustering

Below are some of the clusters that have been found by the K-means algorithm, clustering the 5,000 nouns into 700 clusters[2]:

- april januari november februari oktober maart mei juni augustus december september juli
- sprinter schaatser coureur speelster middenvelder vedette wielrenner aanvoerder keeper landgenoot speler atlete aanvaller renner verdediger atleet kopman bokser voetballer zwemmer spits tennisser doelman
- dollar ton euro meter kilometer kilo pond gulden centimeter
- hoofdredacteur chef commandant secretaris-generaal bestuursvoorzitter bevelhebber hoofdofficier directeur
- maand zomer winter winters week eeuw herfst
- stijger stijgers daler kanshebber winnaars boosdoener verliezer troef verliezers
- bisschop priesters Kerk predikant priester kerk dominee gelovigen kerken bisschoppen

3.1.1 adjective clustering

Next to clustering nouns according to the collocating adjectives, the approach has also been turned upside down: the 5,000 most frequent adjectives have been clustered according to the 20,000 most frequent collocating nouns. Such kind of clustering produces results of the kind below:

- geel paars zwart groen blauw grijs oranje bruin roze wit rood
- Duits Amerikaans Zweeds Russisch buitenlands Brits Belgisch Nederlands Frans Engels Japans Zwitsers Italiaans Spaans Chinees
- rk roomskatholieke russisch-orthodoxe servisch-orthodoxe oudkatholiek
  anglicaans r.k. koptisch Koptisch grieks-orthodoxe
- zonovergoten herfstig winters druilerig zomers regenachtig zonnig
- deplorabel abominabel erbarmelijk penibel mensonterend mensonwaardig benard miserabel

[2] Note that the corpus has been lemmatized, but due to errors of the lemmatizer, tokens might sometimes end up in the clusters.

4 Agglomerative Clustering

Agglomerative clustering also yields some remarkable results. What is remarkable is that the upper nodes present broad semantic categories, such as persons, objects, abstract entities, … Figure 2 shows part of an example of an agglomerative tree, grouping together nouns that designate a time entity. Note that the edges in the tree should not be interpreted as actual 'is a'-relations (as is the case in Wordnet), but rather as adjacency relations.

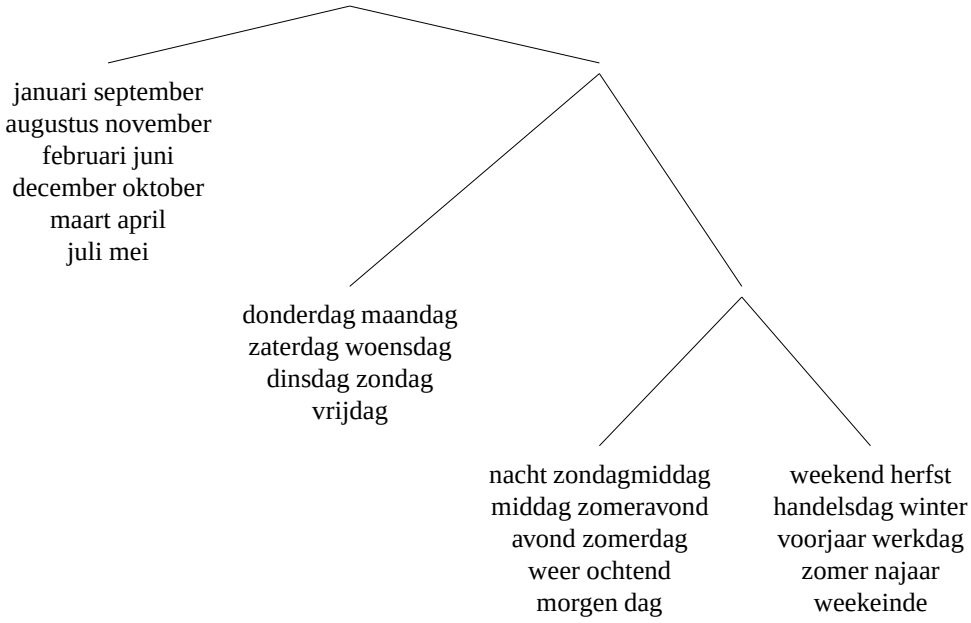

Figure 2: Example of agglomerative clustering: days, months, seasons

5 Evaluation

5.1 Automatic Evaluation with EuroWordNet

5.1.1 Methodology

For the evaluation of the clusters, precision and recall has been calculated according to the relations that exist in EuroWordNet. The procedure of the evaluation is as follows:

- The wordnet relations that are used for the evaluation are:
  - synonyms

  - hyponyms
  - hypernyms
  - co-hyponyms (hyponyms of the hypernyms)

- For each cluster, it is checked in Wordnet which word from the cluster has most relations in Wordnet with the other words from the cluster. This word is taken to be the most central word of the cluster.
- For this word, the synonyms, hyponyms, hypernyms and co-hyponyms are drawn from Wordnet.
- To calculate precision, it is checked how many words from the cluster are actually appearing in the Wordnet-relations.
- To calculate recall, it is checked how many of the Wordnet-relations are not appearing in the found cluster.

A few remarks are to be made with regard to this evaluation framework. Recall will be a low number, because different kind of relations are considered in the evaluation framework: hyponyms, hypernyms, synonyms and co-hyponyms are considered all together. The real recall (as acknowledged by human judges) is probably a lot higher. To a human judge, a cluster that contains the 7 days of the week seems quite complete, but in this evaluation framework, it gets a recall of 9.21%. Therefore, recall is not such a good measure in this evaluation framework; precision will therefore be considered the most important value.

5.1.2 Results

Table 1 gives the results of the evaluation of the **partitional** clustering algorithm of **nouns**. The precision and recall values are given according to the number of clusters used. Figure 3 presents these results graphically.

The figures show a precision that is lower with few, large clusters, rising towards an optimal number of clusters, and then declining again when the number of clusters gets too small. Recall is faintly showing the opposite tendency, being larger with fewer but large clusters, and declining when the number of clusters get larger. But as has been explained before, recall is not such a good measure in this case.

With the optimal number of clusters, the clustering algorithm is able to reach a precision of **42.50%** (and a recall of about **8%**). Taking into account that Wordnet itself is incomplete (a large part of the clustered words is not known by Wordnet), that the evaluation algorithm is only looking one level up and down in the wordnet hierarchy, and that there is an error margin due to mistakes of the lemmatizer, the results obtained by the clustering algorithm are quite good. The random baseline (results for clusters that have been randomly compiled using a hash table) is about **5.5%** for precision and about **3%** for recall.

# clusters	precision	recall	# clusters	precision	recall
200	31.14%	7.83%	1200	42.50%	6.82%
300	33.94%	6.80%	1300	42.12%	7.23%
400	35.84%	6.85%	1400	42.48%	7.59%
500	38.00%	8.12%	1500	41.48%	7.80%
600	39.62%	7.59%	1600	40.68%	7.00%
700	39.76%	8.19%	1700	40.04%	7.24%
800	40.92%	8.15%	1800	38.32%	7.56%
900	41.18%	7.94%	1900	37.32%	7.34%
1000	42.62%	7.20%	2000	35.40%	7.27%
1100	42.14%	7.49%	2500	28.40%	4.99%
			random	5.54%	2.75%

Table 1: Evaluation of 5000 nouns clustering

It is interesting to have a look at the share of each relationship in the precision measure. This gives an indication of the relationships that are found by the clustering algorithm, and to what extent they are found. Table 2 gives the results of this subdivision, and figure 4 shows the share of each relationship graphically.

The majority of words found by the clustering algorithm are clearly co-hyponyms. This result was to be expected, as the horizontal relationship (which is mainly the co-hyponym relationship) is the relationship in which most similarity is to be found. Synonyms (which is the second horizontal relationship) are also found by the algorithm, but the share of synonyms is of the same order as the hyponym and hypernym relationships. Most likely, this is because synonyms are less numerous than co-hyponyms. What is remarkable, is that co-hyponyms and synonyms seem to be following the same pattern: starting at a lower precision with large (fewer) clusters, rising to an optimum with middle-size clusters and declining again when the clusters get too small (too many clusters). This is not the case with the other relationships: the hypernym precision stays roughly at the same level (and is even rising a bit), while the hyponym precision is declining with the number of clusters. These results seem to indicate that hyponyms tend to get clustered more easily than hypernyms, when the margin is large enough (fewer but larger clusters). Of course, these tendencies are not significant enough to draw conclusions.

5.2 Evaluation with Wu & Palmer's measure

The algorithm discussed in 5.1 tries to evaluate the cluster quality by comparing the clusters to a fixed set of words extracted from EuroWordNet. This makes it possible to calculate precision and recall values. Another kind of evaluation relies on measures of **semantic similarity** according to hierarchical wordnets.

A number of such similarity measures have been developed. Among these

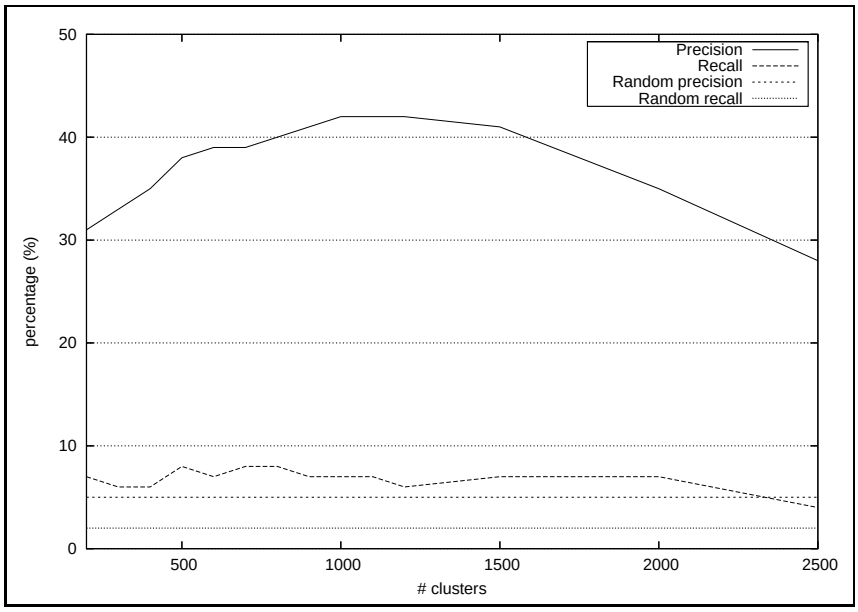

Figure 3: Evaluation of 5,000 nouns clustering with Wordnet

measures, the most important are Wu & Palmer's (Wu and Palmer 1994), Resnik's (Resnik 1995) and Lin's (Lin 1998).

In this evaluation, Wu & Palmer's measure will be adopted. Wu and Palmer (1994) have proposed a measure that calculates the similarity between two words according to their position in a hierarchical wordnet. The similarity is calculated according to the formula in 5.2, in which N_1 and N_2 are the number of *is-a* links from A and B to their most specific common superclass C; N_3 is the number of *is-a* links from C to the root of the taxonomy.

$$sim_{Wu\&Palmer}(A, B) = \frac{2N_3}{N_1 + N_2 + 2N_3}$$

For example, the most common superclass of *hond* en *zalm* is *dier* (as can be seen on the extract from Dutch EuroWordNet in figure 5). Consequently, $N_1 = 2$, $N_2 = 2$, $N_3 = 4$ and $sim_{Wu\&Palmer}(hond, zalm) = 0.67$.

The results have not been calculated by using the Dutch EuroWordNet directly, as was the case with the former evaluation framework. Instead, the words have been converted to Interlingual WordNet offsets. This way, it was possible to make use of a perl module that implements the computation of Wu & Palmer's measure in the English Wordnet (Pedersen, Patwardhan and Michelizzi 2004).

The average cluster similarity has been calculated as follows:

- For each cluster, the most central word has been taken (which is the word that was the closest to the cluster's centroid);

# clusters	synonyms	hyponyms	hypernyms	co-hyponyms	total
200	2.58%	4.58%	1.76%	22.22%	31.14%
300	3.00%	5.18%	2.30%	23.46%	33.94%
400	3.82%	5.74%	2.58%	23.70%	35.84%
500	4.64%	5.78%	2.82%	24.76%	38.00%
600	4.94%	6.52%	3.04%	25.12%	39.62%
700	5.40%	5.86%	2.82%	25.68%	39.76%
800	5.88%	5.84%	3.12%	26.08%	40.92%
900	6.12%	5.78%	3.32%	25.96%	41.18%
1000	6.50%	6.06%	2.94%	27.12%	42.62%
1100	6.44%	5.54%	3.38%	26.78%	42.14%
1200	6.40%	5.84%	3.36%	26.90%	42.50%
1300	6.66%	5.02%	3.36%	27.08%	42.12%
1400	6.60%	5.36%	4.00%	26.52%	42.48%
1500	6.58%	4.84%	3.58%	26.48%	41.48%
1600	6.98%	4.62%	3.52%	25.56%	40.68%
1700	6.72%	4.32%	3.78%	25.22%	40.04%
1800	6.38%	4.66%	3.34%	23.94%	38.32%
1900	6.20%	4.12%	3.56%	23.44%	37.32%
2000	6.12%	3.64%	3.36%	22.28%	35.40%
2500	4.70%	2.62%	2.88%	18.20%	28.40%

Table 2: Share of each relationship in total precision

- the similarity between this most central word and every other word in the cluster has been calculated;
- the average of these similarities has been taken, and every cluster average has been added up;
- all cluster averages have been divided by the total number of clusters, to get the total average;
- words not known by Wordnet have been ignored.

Table 3 gives the results of the evaluation with Wu&Palmer's measure, and figure 6 shows the results graphically. There's an average of 60% similarity within the clusters, while randomly compiled clusters have an average of 29% similarity. These results seem to confirm the results found by the former evaluation framework.

6 Conclusion & Further Work

In this research, machine learning techniques have been explored that make it possible to automatically acquire semantic classes in Dutch. More particularly, vector

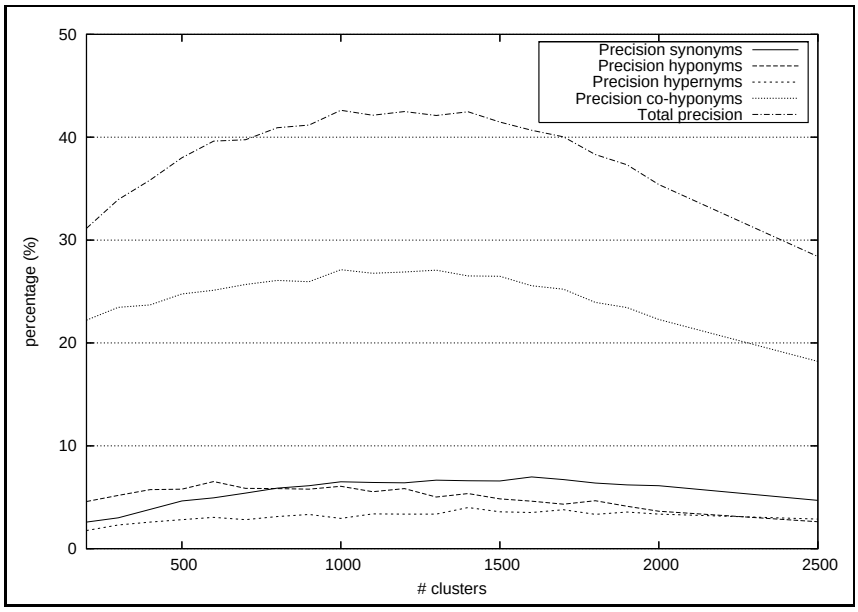

Figure 4: Share of each relationship in total precision

space measures have been used to calculate semantic similarity. These semantic similarity measures are then used to cluster nouns into classes. Partitional K-means clustering has been used as a partitional clustering algorithm and group-average agglomerative clustering has been used as a hierarchical clustering algorithm.

The results and the evaluation of the clusters have shown that using the syntactic context of words (more particularly, using modifiers to cluster nouns) is indeed a good approach for extracting semantic classes. The evaluation of the clusters shows significant similarities with Wordnet-relations, in my own evaluation framework evaluating direct relationships, as well as with Wu & Palmer's similarity measure. The clustering of modifiers (adjectives) according to heir heads (nouns) also seems to yield quite good results, although this kind of clustering has

# clusters	similarity	# clusters	similarity
500	54.69%	1200	59.45%
700	55.73%	1500	60.40%
800	56.78%	2000	59.51%
900	57.92%	2500	59.44%
1000	58.54%	*random*	29.00%

Table 3: Evaluation of 5,000 nouns clustering (with Wu & Palmer's similarity measure)

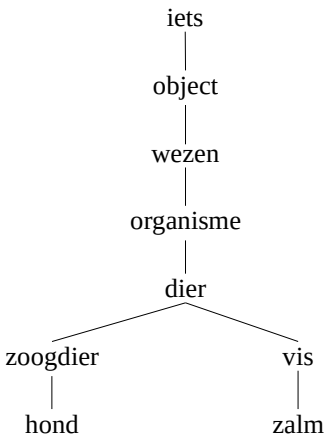

Figure 5: Extract from the Dutch EuroWordNet Hierarchy

not yet been evaluated.

There are, however, some issues that make the automatic clustering of nouns less straightforward. Ambiguity is one of the problems that is difficult to tackle for a computer. Ambiguity blurs the results, because both senses of the word (with their accompanying values) get grouped together into one sense. Disambiguating these ambiguous words into different clusters is one of the main goals to be solved, in order to reach a semantic clustering that is able to compete with hand-crafted semantic classes.

Another issue that requires more research is the automatic extraction of complete wordnets, instead of stand-alone clusters. Instead of focusing on the horizontal semantic relationships (creating clusters of similar words) it would be interesting to explore algorithms that automatically acquire the vertical relationships. This way, it might be possible to automatically construct a complete wordnet, similar to hand-crafted wordnets available. The agglomerative clustering algorithm is a first step towards this direction, though still far from perfect.

Also, the field of verb clustering remains to be explored. Subject-verb and verb-object relations are quite different from adjective-noun relations. How these relations might be used in order to cluster verbs, is subject to further research. It also remains to be investigated how these relations might help in improving the clustering of nouns.

A final interesting subject for future research is the application of dimensionality reduction techniques (LSA, PLSA) to counter data sparseness and noise. The application of these techniques will be explored in order to bring about a better clustering.

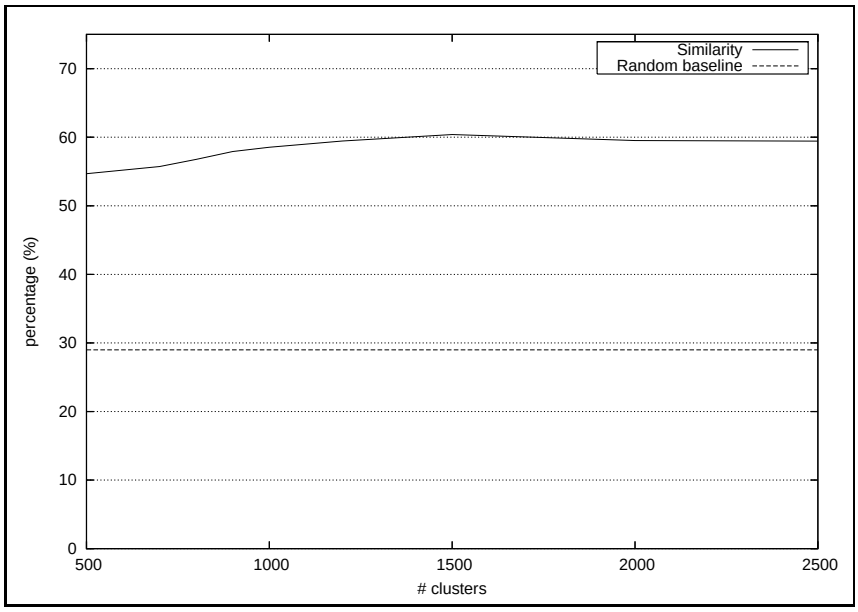

Figure 6: Evaluation the clustering of nouns (Wu&Palmer similarity measure)

Acknowledgements

This research has been carried out during a 3 month internship at the *Centrum voor Nederlandse Taal en Spraak* at the University of Antwerp. The work has been supervised by Walter Daelemans and Bart Decadt, and has substantially benefited from their remarks and contributions.

References

Daelemans, W., Zavrel, J. et al.(1996), Mbt: A memory-based part of speech tagger-generator, *in* E. Ejerhed and I. Dagan (eds), *Proceedings of the Fourth Workshop on Very Large Corpora*, Copenhagen, Denmark, pp. 14–27.

Harris, Z.(1985), Distributional structure, *in* J. J. Katz (ed.), *The Philosophy of Linguistics*, Oxford University Press, pp. 26–47.

Lin, D.(1998), An information-theoretic definition of similarity, *Proceedings of the 15th International Conference on Machine Learning*, Morgan Kaufmann, San Francisco, pp. 296–304.

Manning, C. and Schütze, H.(2000), *Foundations of Statistical Natural Language Processing*, MIT Press, Cambridge, Massachussets.

Pantel, P. and Lin, D.(2002), Discovering word senses from text.

Pedersen, T., Patwardhan, S. and Michelizzi, J.(2004), Wordnet::similarity - measuring the relatedness of concepts.

Rennie, J.(2000), Wordnet::querydata: a Perl module for accessing the WordNet database, http://people.csail.mit.edu/ jrennie/WordNet.

Resnik, P.(1995), Using information content to evaluate semantic similarity in a taxonomy, *IJCAI*, pp. 448–453.

van den Bosch, A. and Daelemans, W.(1999), Memory-based morphological analysis, *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, University of Maryland, USA, pp. 285–292.

Wu, Z. and Palmer, M.(1994), Verb semantics and lexical selection, *32nd. Annual Meeting of the Association for Computational Linguistics*, New Mexico State University, Las Cruces, New Mexico, pp. 133 –138.