Using Three Way Data for Word Sense Discrimination

Tim Van de Cruys
Humanities Computing
University of Groningen
t.van.de.cruys@rug.nl

Abstract

In this paper, an extension of a dimensionality reduction algorithm called NON-NEGATIVE MATRIX FACTORIZATION is presented that combines both 'bag of words' data and syntactic data, in order to find semantic dimensions according to which both words and syntactic relations can be classified. The use of three way data allows one to determine which dimension(s) are responsible for a certain sense of a word, and adapt the corresponding feature vector accordingly, 'subtracting' one sense to discover another one. The intuition in this is that the syntactic features of the syntax-based approach can be disambiguated by the semantic dimensions found by the bag of words approach. The novel approach is embedded into clustering algorithms, to make it fully automatic. The approach is carried out for Dutch, and evaluated against EuroWordNet.

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 (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.' (Manning and Schütze, 2000, §8.5)

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. With regard to the context used, two basic approaches exist. One approach makes use of 'bag of words' co-occurrence data; in this approach, a certain window around a word is used for gathering co-occurrence information. The window may either be a fixed number of words, or the paragraph or document that a word appears in. Thus, words are considered similar if they appear in similar windows (documents). One of the dominant methods using this method is LATENT SEMANTIC ANALYSIS (LSA).

The second approach uses a more fine grained distributional model, focusing on the syntactic relations that words appear with. Typically, a large text corpus is parsed, and dependency triples are extracted.[1] Words are considered similar if they appear with similar syntactic relations. Note that the former approach does not need any kind of linguistic annotation, whereas for the latter, some form of syntactic annotation is needed.

The results yielded by both approaches are typically quite different in nature: the former approach typically puts its finger on a broad, thematic kind of similarity, while the latter approach typically grasps a tighter, synonym-like similarity. Example (1) shows the difference between both approaches; for each approach, the top ten most similar nouns to the Dutch noun *muziek* 'music' are given. In (a), the window-based approach is used, while (b) uses the syntax-based approach. (a) shows indeed more thematic similarity, whereas (b) shows tighter similarity.

[1] e.g. dependency relations that qualify *apple* might be 'object of *eat*' and 'adjective *red*'. This gives us dependency triples like $< apple, obj, eat >$.

Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), pages 929–936
Manchester, August 2008

(1) a. **muziek** 'music': *gitaar* 'guitar', *jazz* 'jazz', *cd* 'cd', *rock* 'rock', *bas* 'bass', *song* 'song', *muzikant* 'musician', *musicus* 'musician', *drum* 'drum', *slagwerker* 'drummer'

b. **muziek** 'music': *dans* 'dance', *kunst* 'art', *klank* 'sound', *liedje* 'song', *geluid* 'sound', *poëzie* 'poetry', *literatuur* 'literature', *popmuziek* 'pop music', *lied* 'song', *melodie* 'melody'

Especially the syntax-based method has been adopted by many researchers, in order to find semantically similar words. There is, however, one important problem with this kind of approach: the method is not able to cope with ambiguous words. Take the examples:

(2) een oneven nummer
a odd number
an odd number

(3) een steengoed nummer
a great number
'a great song'

The word *nummer* does not have the same meaning in these examples. In example (2), *nummer* is used in the sense of 'designator of quantity'. In example (3), it is used in the sense of 'musical performance'. Accordingly, we would like the word *nummer* to be disambiguated into two senses, the first sense being similar to words like *getal* 'number', *cijfer* 'digit' and the second to words like *liedje* 'song', *song* '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. Even worse: the results get blurred because the attributes of both senses (in this example *oneven* and *steengoed*) are grouped together into one sense. This is the main drawback of the syntax-based method. On the other hand, methods that capture semantic dimensions are known to be useful in disambiguating different senses of a word. Particularly, PROBABILISTIC LATENT SEMANTIC ANALYSIS (PLSA) is known to simultaneously encode various senses of words according to latent semantic dimensions (Hofmann, 1999). In this paper, we want to explore an approach that tries to remedy the shortcomings of the former, syntax-based approach with the benefits of the latter. The intuition in this is that the syntactic features of the syntax-based approach can be disambiguated by the 'latent semantic dimensions' found by the window-based approach.

2 Previous Work

2.1 Distributional Similarity

There have been numerous approaches for computing the similarity between words from distributional data. We mention some of the most important ones.

With regard to the first approach – using a context window – we already mentioned LSA (Landauer and Dumais, 1997). In LSA, a term-document matrix is created, containing the frequency of each word in a specific document. This matrix is then decomposed into three other matrices with a mathematical technique called SINGULAR VALUE DECOMPOSITION. The most important dimensions that come out of the SVD allegedly represent 'latent semantic dimensions', according to which nouns and documents can be presented more efficiently.

LSA has been criticized for not being the most appropriate data reduction method for textual applications. The SVD underlying the method assumes normally-distributed data, whereas textual count data (such as the term-document matrix) can be more appropriately modeled by other distributional models such as Poisson (Manning and Schütze, 2000, §15.4.3). Successive methods such as PROBABILISTIC LATENT SEMANTIC ANALYSIS (PLSA) (Hofmann, 1999), try to remedy this shortcoming by imposing a proper latent variable model, according to which the values can be estimated. The method we adopt in our research – NON-NEGATIVE MATRIX FACTORIZATION – is similar to PLSA, and adequately remedies this problem as well.

The second approach – using syntactic relations – has been adopted by many researchers, in order to acquire semantically similar words. One of the most important is Lin's (1998). For Dutch, the approach has been applied by Van der Plas & Bouma (2005).

2.2 Discriminating senses

Schütze (1998) uses a disambiguation algorithm – called context-group discrimination – based on the clustering of the context of ambiguous words. The clustering is based on second-order co-occurrence: the contexts of the ambiguous word are similar if the words they in turn co-occur with are similar.

Pantel and Lin (2002) present a clustering algorithm – coined CLUSTERING BY COMMITTEE (CBC) – that automatically discovers word senses from text. The key idea is to first discover a set of tight, unambiguous clusters, to which possibly ambiguous words can be assigned. Once a word has been assigned to a cluster, the features associated with that particular cluster are stripped off the word's vector. This way, less frequent senses of the word can be discovered.

The former approach uses a window-based method; the latter uses syntactic data. But none of the algorithms developed so far have combined both sources in order to discriminate among different senses of a word.

3 Methodology

3.1 Non-negative Matrix Factorization

3.1.1 Theory

Non-negative matrix factorization (NMF) (Lee and Seung, 2000) is a group of algorithms in which a matrix V is factorized into two other matrices, W and H.

$$V_{n \times m} \approx W_{n \times r} H_{r \times m} \tag{1}$$

Typically r is much smaller than n, m so that both instances and features are expressed in terms of a few components.

Non-negative matrix factorization enforces the constraint that all three matrices must be non-negative, so all elements must be greater than or equal to zero. The factorization turns out to be particularly useful when one wants to find additive properties.

Formally, the non-negative matrix factorization is carried out by minimizing an objective function. Two kinds of objective function exist: one that minimizes the Euclidean distance, and one that minimizes the Kullback-Leibler divergence. In this framework, we will adopt the latter, as – from our experience – entropy-based measures tend to work well for natural language. Thus, we want to find the matrices W and H for which the Kullback-Leibler divergence between V and WH (the multiplication of W and H) is the smallest.

Practically, the factorization is carried out through the iterative application of update rules. Matrices W and H are randomly initialized, and the rules in 2 and 3 are iteratively applied – alternating between them. In each iteration, each vector is adequately normalized, so that all dimension values sum to 1.

$$H_{a\mu} \leftarrow H_{a\mu} \frac{\sum_i W_{ia} \frac{V_{i\mu}}{(WH)_{i\mu}}}{\sum_k W_{ka}} \tag{2}$$

$$W_{ia} \leftarrow W_{ia} \frac{\sum_\mu H_{a\mu} \frac{V_{i\mu}}{(WH)_{i\mu}}}{\sum_v H_{av}} \tag{3}$$

3.1.2 Example

We can now straightforwardly apply NMF to create semantic word models. NMF is applied to a frequency matrix, containing bag of words co-occurrence data. The additive property of NMF ensures that semantic dimensions emerge, according to which the various words can be classified. Two sample dimensions are shown in example (4). For each dimension, the words with the largest value on that dimension are given. Dimension (a) can be qualified as a 'transport' dimension, and dimension (b) as a 'cooking' dimension.

(4) a. *bus* 'bus', *taxi* 'taxi', *trein* 'train', *halte* 'stop', *reiziger* 'traveler', *perron* 'platform', *tram* 'tram', *station* 'station', *chauffeur* 'driver', *passagier* 'passenger'

b. *bouillon* 'broth', *slagroom* 'cream', *ui* 'onion', *eierdooier* 'egg yolk', *laurierblad* 'bay leaf', *zout* 'salt', *deciliter* 'decilitre', *boter* 'butter', *bleekselderij* 'celery', *saus* 'sauce'

3.2 Extending Non-negative Matrix Factorization

We now propose an extension of NMF that combines both the bag of words approach and the syntactic approach. The algorithm finds again latent semantic dimensions, according to which nouns, contexts and syntactic relations are classified.

Since we are interested in the classification of nouns according to both 'bag-of-words' context and syntactic context, we first construct three matrices that capture the co-occurrence frequency information for each mode. The first matrix contains co-occurrence frequencies of nouns cross-classified by dependency relations, the second matrix contains co-occurrence frequencies of nouns cross-classified by words that appear in the noun's context window, and the third matrix contains co-occurrence frequencies of dependency relations cross-classified by co-occurring context words.

We then apply NMF to the three matrices, but we interleave the separate factorizations: the results of the former factorization are used to initialize the factorization of the next matrix. This implies that we need to initialize only three matrices at random; the other three are initialized by calculations of the previous step. The process is represented graphically in figure 1.

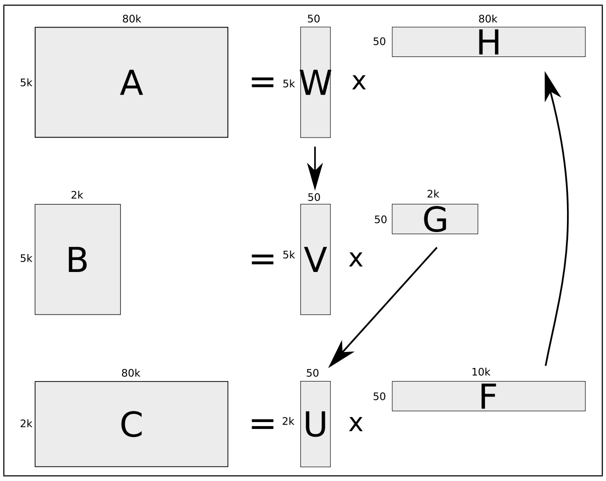

Figure 1: A graphical representation of the extended NMF

In the example in figure 1, matrix H is initialized at random, and the update of matrix W is calculated. The result of update W is then used to initialize matrix V, and the update of matrix G is calculated. This matrix is used again to initialize matrix U, and the update of matrix F is calculated. This matrix can be used to initialize matrix H, and the process is repeated until convergence.

In (5), an example is given of the kind of semantic dimensions found. This dimension may be coined the 'transport' dimension, as is shown by the top 10 nouns (a), context words (b) and syntactic relations (c).

(5)
a. *auto* 'car', *wagen* 'car', *tram* 'tram', *motor* 'motorbike', *bus* 'bus', *metro* 'subway', *automobilist* 'driver', *trein* 'trein', *stuur* 'steering wheel', *chauffeur* 'driver'
b. *auto* 'car', *trein* 'train', *motor* 'motorbike', *bus* 'bus', *rij* 'drive', *chauffeur* 'driver', *fiets* 'bike', *reiziger* 'reiziger', *passagier* 'passenger', *vervoer* 'transport'
c. viertraps$_{adj}$ 'four pedal', verplaats_met$_{obj}$ 'move with', toeter$_{adj}$ 'honk', tank_in_houd$_{obj}$ [parsing error], tank$_{subj}$ 'refuel', tank$_{obj}$ 'refuel', rij_voorbij$_{subj}$ 'pass by', rij_voorbij$_{adj}$ 'pass by', rij_af$_{subj}$ 'drive off', peperduur$_{adj}$ 'very expensive'

3.3 Sense Subtraction

Next, we want to use the factorization that has been created in the former step for word sense discrimination. The intuition is that we 'switch off' one dimension of an ambiguous word, to reveal possible other senses of the word. From matrix H, we know the importance of each syntactic relation given a dimension. With this knowledge, we can 'subtract' the syntactic relations that are responsible for a certain dimension from the original noun vector:

$$\overrightarrow{v}_{new} = \overrightarrow{v}_{orig}(\overrightarrow{1} - \overrightarrow{h}_{dim}) \qquad (4)$$

Equation 4 multiplies each feature (syntactic relation) of the original noun vector ($\overrightarrow{v}_{orig}$) with a scaling factor, according to the load of the feature on the subtracted dimension ($\overrightarrow{h}_{dim}$ – the vector of matrix H containing the dimension we want to subtract). $\overrightarrow{1}$ is a vector of ones, the size of $\overrightarrow{h}_{dim}$.

3.4 A Clustering Framework

The last step is to determine which dimension(s) are responsible for a certain sense of the word. In order to do so, we embed our method in a clustering approach. First, a specific word is assigned to its predominant sense (i.e. the most similar cluster). Next, the dominant semantic dimension(s) for this cluster are subtracted from the word vector (equation 4), and the resulting vector is fed to the clustering algorithm again, to see if other word senses emerge. The dominant semantic dimension(s) can be identified by 'folding in' the cluster centroid into our factorization (so we get a vector $\overrightarrow{w}$ of dimension size r), and applying a threshold to the result (in our experiments a threshold of $\delta = .05$ — so dimensions responsible for $> 5\%$ of the centroid are subtracted).

We used two kinds of clustering algorithms to determine our initial centroids. The first algorithm is a standard K-means algorithm. The second one is the CBC algorithm by Pantel and Lin (2002). The initial vectors to be clustered are adapted with pointwise mutual information (Church and Hanks, 1990).

3.4.1 K-means

First, a standard K-means algorithm is applied to the nouns we want to cluster. This yields a hard clustering, in which each noun is assigned to exactly one (dominant) cluster. In the second step, we try to determine for each noun whether it can be assigned to other, less dominant clusters. First, the salient dimension(s) of the centroid to which the noun is assigned are determined. We compute the centroid of the cluster by averaging the frequencies of all cluster elements except for the target element we want to reassign, and adapt the centroid with pointwise mutual information. After subtracting the salient dimensions from the noun vector, we check whether the vector is reassigned to another cluster centroid (i.e. whether it is more similar to a different centroid). If this is the case, (another instance of) the noun is assigned to the cluster, and we repeat the second step. If there is no reassignment, we continue with the next word. The target element is removed from the centroid to make sure that we only subtract the dimensions associated with the sense of the cluster.

Note that K-means requires to set the number of clusters beforehand, so k is a parameter to be set.

3.4.2 CBC

The second clustering algorithm operates in a similar vein, but instead of using simple K-means, we use Pantel and Lin's CBC algorithm to find the initial centroids (coined COMMITTEES).

In order to find committees, the top k nouns for each noun in the database are clustered with average-link clustering. The clusters are scored and sorted in such a way that preference is given to tight, representative clusters. If the committees do not cover all elements sufficiently, the algorithm recursively tries to find more committees. An elaborate description of the algorithm can be found in (Pantel and Lin, 2002).

In the second step, we start assigning elements to committees. Once an element is assigned, the salient dimensions are subtracted from the noun vector in the same way as in 3.4.1 (only do we not have to remove any target word from the centroid; committees are supposed to represent tight, unambiguous clusters).

CBC attempts to find the number of committees automatically from the data, so k does not have to be set.

4 Examples

4.1 Sense Subtraction

In what follows, we will talk about semantic dimensions as, e.g., the 'music' dimension or the 'city' dimension. In the vast majority of the cases, the dimensions are indeed as clear-cut as the transport dimension shown above, so that the dimensions can be rightfully labeled this way.

Two examples are given of how the semantic dimensions that have been found can be used for word sense discrimination. We will consider two ambiguous nouns: *pop*, which can mean 'pop music' as well as 'doll', and *Barcelona*, which can designate either the Spanish city or the Spanish football club.

First, we look up the top dimensions for each noun. Next, we successively subtract the dimensions dealing with a particular sense of the noun, as described in 3.3. This gives us three vectors for each noun: the original vector, and two vectors with one of the dimensions eliminated. For each of these vectors, the top ten similar nouns are given, in order to compare the changes brought about.

(6) a. *pop, rock, jazz, meubilair* 'furniture', *popmuziek* 'pop music', *heks* 'witch', *speelgoed* 'toy', *kast* 'cupboard', *servies* '[tea] service', *vraagteken* 'question mark'
b. *pop, meubilair* 'furniture', *speelgoed* 'toy', *kast* 'cupboard', *servies* '[tea] service', *heks* 'witch', *vraagteken* 'question mark' *sieraad* 'jewel', *sculptuur* 'sculpture', *schoen* 'shoe'
c. *pop, rock, jazz, popmuziek* 'pop music', *heks* 'witch', *danseres* 'dancer', *servies* '[tea] service', *kopje* 'cup', *house* 'house music', *aap* 'monkey'

Example (6) shows the top similar words for the three vectors of *pop*. In (a), the most similar words to the original vector are shown. In (b), the top dimension (the 'music dimension') has been subtracted from (a), and in (c), the second highest dimension (a 'domestic items' dimension) has been subtracted from (a).

The differences between the three vectors are clear: in vector (a), both senses are mixed together, with 'pop music' and 'doll' items interleaved. In (b), no more music items are present. Only items related to the doll sense are among the top similar words. In (c), the music sense emerges much more clearly, with *rock*, *jazz* and *popmuziek* being the most similar, and a new music term (*house*) showing up among the top ten.

Admittedly, in vector (c), not all items related to the 'doll' sense are filtered out. We believe this is due to the fact that this sense cannot be adequately filtered out by one dimension (in this case, a dimension of 'domestic items' alone), whereas it is much easier to filter out the 'music' sense with only one 'music' dimension. We will try to remedy this in our clustering framework, in which it is possible to subtract multiple dimensions related to one sense.

A second example, the ambiguous proper name *Barcelona*, is given in (7).

(7) a. *Barcelona, Arsenal, Inter, Juventus, Vitesse, Milaan* 'Milan', *Madrid, Parijs* 'Paris', *Wenen* 'Vienna', *München* 'Munich'
b. *Barcelona, Milaan* 'Milan', *München* 'Mu-

nich’, *Wenen* ‘Vienna’, *Madrid, Parijs* ‘Paris’, *Bonn, Praag* ‘Prague’, *Berlijn* ‘Berlin’, *Londen* ‘London’

c. *Barcelona, Arsenal, Inter, Juventus, Vitesse, Parma, Anderlecht, PSV, Feyenoord, Ajax*

In (a), the two senses of *Barcelona* are clearly mixed up, showing cities as well as football clubs among the most similar nouns. In (b), where the ‘football dimension’ has been subtracted, only cities show up. In (c), where the ‘city dimension’ has been subtracted, only football clubs remain.

4.2 Clustering Output

In (8), an example of our clustering algorithm with initial K-means clusters is given.

(8) a. *werk* ‘work’ *beeld* ‘image’ *foto* ‘photo’ *schilderij* ‘painting’ *tekening* ‘drawing’ *doek* ‘canvas’ *installatie* ‘installation’ *afbeelding* ‘picture’ *sculptuur* ‘sculpture’ *prent* ‘picture’ *illustratie* ‘illustration’ *handschrift* ‘manuscript’ *grafiek* ‘print’ *aquarel* ‘aquarelle’ *maquette* ‘scale-model’ *collage* ‘collage’ *ets* ‘etching’

b. *werk* ‘work’ *boek* ‘book’ *titel* ‘title’ *roman* ‘novel’ *boekje* ‘booklet’ *debuut* ‘debut’ *biografie* ‘biography’ *bundel* ‘collection’ *toneelstuk* ‘play’ *bestseller* ‘bestseller’ *kinderboek* ‘child book’ *autobiografie* ‘autobiography’ *novelle* ‘short story’

c. *werk* ‘work’ *voorziening* ‘service’ *arbeid* ‘labour’ *opvoeding* ‘education’ *kinderopvang* ‘child care’ *scholing* ‘education’ *huisvesting* ‘housing’ *faciliteit* ‘facility’ *accommodatie* ‘acommodation’ *arbeidsomstandigheid* ‘working condition’

The example shows three different clusters to which the noun *werk* ‘work’ is assigned. In (a), *werk* refers to a work of art. In (b), it refers to a written work. In (c), the ‘labour’ sense of *werk* emerges.

5 Evaluation

5.1 Methodology

The clustering results are evaluated according to Dutch EuroWordNet (Vossen and others, 1999). Precision and recall are calculated by comparing the results to EuroWordNet synsets. The precision is the number of clusters found that correspond to an actual sense of the word. Recall is the number of word senses in EuroWordNet that are found by the algorithm. Our evaluation method is largely the same as the one used by Pantel and Lin (2002).

Both precision and recall are based on wordnet similarity. A number of similarity measures have been developed to calculate semantic similarity in a hierarchical wordnet. Among these 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 (1994) measure will be adopted. The similarity is calculated according to the formula in (5), 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.

$$sim_{Wu\&Palmer}(A,B) = \frac{2N_3}{N_1 + N_2 + 2N_3} \quad (5)$$

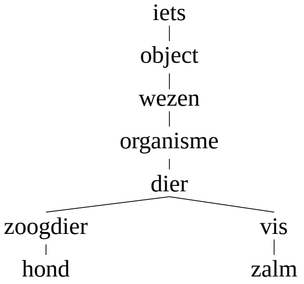

Figure 2: Extract from the Dutch EuroWordNet Hierarchy

For example, the most common superclass of *hond* ‘dog’ en *zalm* ‘salmon’ is *dier* ‘animal’ (as can be seen on the extract from Dutch EuroWordNet in figure 2). Consequently, $N_1 = 2$, $N_2 = 2$, $N_3 = 4$ and $sim_{WP}(hond, zalm) = 0.67$.

To calculate precision, we apply the same methodology as Pantel and Lin (2002).[2] Let $S(w)$ be the set of EuroWordNet senses. $sim_W(s,u)$, the similarity between a synset s and a word u is then defined as the maximum similarity between s and a sense of u:

$$sim_W(s,u) = \max_{t \epsilon S(u)} sim(s,t) \quad (6)$$

Let c_k be the top k-members of a cluster c, where these are the k most similar members to the centroid of c. $simC(c,s)$, the similarity between s and c, is then defined as the average similarity between s and the top-k members of c:

$$sim_C(s,c) = \frac{\sum_{u \epsilon c_k} simW(s,u)}{k} \quad (7)$$

[2] Note, however, that our similarity measure is different. Where Pantel and Lin use Lin’s (1998) measure, we use Wu and Palmer’s (1994) measure.

An assigment of a word w to a cluster c can now be classified as correct if

$$\max_{s \epsilon S(w)} simC(s, c) > \theta \qquad (8)$$

and the EuroWordNet sense of w that corresponds to c is

$$\arg\max_{s \epsilon S(w)} simC(s, c) \qquad (9)$$

When multiple clusters correspond to the same EuroWordNet sense, only one of them is counted as correct.

Precision of a word w is the percentage of correct clusters to which it is assigned. Recall of a word w is the percentage of senses from EuroWordnet that have a corresponding cluster.[3] Precision and recall of a clustering algorithm is the average precision and recall of all test words.

5.2 Experimental Design

We have applied the interleaved NMF presented in section 3.2 to Dutch, using the TWENTE NIEUWS CORPUS (Ordelman, 2002), containing $>$ 500M words of Dutch newspaper text. The corpus is consistently divided into paragraphs, which have been used as the context window for the bag-of-words mode. The corpus has been parsed by the Dutch dependency parser Alpino (van Noord, 2006), and dependency triples have been extracted. Next, the three matrices needed for our method have been constructed: one containing nouns by dependency relations (5K $\times$ 80K), one containing nouns by context words (5K $\times$ 2K) and one containing dependency relations by context words (80K $\times$ 2K). We did 200 iterations of the algorithm, factorizing the matrices into 50 dimensions. The NMF algorithm has been implemented in Matlab.

For the evaluation, we use all the words that appear in our original clustering input as well as in EuroWordNet. This yields a test set of 3683 words.

5.3 Results

Table 1 shows precision and recall figures for four different algorithms, according to two similarity thresholds θ (equation 8). kmeans$_{nmf}$ describes the results of our algorithm with K-means clusters, as described in section 3.4.1. CBC describes the results of our algorithm with the CBC committees, as described in section 3.4.2. For comparison, we have also included the results of a standard K-means clustering (kmeans$_{orig}$, $k = 600$), and the original CBC algorithm (CBC$_{orig}$) as described by Pantel and Lin (2002).

		threshold θ	
		.40 (%)	.60 (%)
kmeans$_{nmf}$	prec.	78.97	55.16
	rec.	63.90	44.77
CBC$_{nmf}$	prec.	82.70	54.87
	rec.	60.27	40.51
kmeans$_{orig}$	prec.	86.13	58.97
	rec.	60.23	41.80
CBC$_{orig}$	prec.	44.94	29.74
	rec.	69.61	48.00

Table 1: Precision and recall for four different algorithms according to two similarity thresholds

The results show the same tendency across all similarity thresholds: kmeans$_{nmf}$ has a high precision, but lower recall compared to CBC$_{orig}$. Still the recall is higher compared to standard K-means, which indicates that the algorithm is able to find multiple senses of nouns, with high precision. The results of CBC$_{nmf}$ are similar to the results of kmeans$_{orig}$, indicating that few words are reassigned to multiple clusters when using CBC committees with our method.

Obviously, kmeans$_{orig}$ scores best with regard to precision, but worse with regard to recall. CBC$_{orig}$ finds most senses (highest recall), but precision is considerably worse.

The fact that recall is already quite high with standard K-means clustering indicates that the evaluation is skewed towards nouns with only one sense, possibly due to a lack of coverage in EuroWordNet. In future work, we specifically want to evaluate the discrimination of ambiguous words. Also, we want to make use of the new Cornetto Database[4], a successor of EuroWordNet for Dutch which is currently under development.

Still, the evaluation shows that our method provides a genuine way of finding multiple senses of words, while retaining high precision. Especially the method using a simple K-means clustering per-

[3] Our notion of recall is slightly different from the one used by Pantel and Lin, as they use 'the number of senses in which w was used in the corpus' as gold standard. This information, as they acknowledge, is difficult to get at, so we prefer to use the sense information in EuroWordNet.

[4] `http://www.let.vu.nl/onderzoek/projectsites/cornetto/index.html`

forms particularly well. The three way data allows the algorithm to put its finger on the particular sense of a centroid, and adapt the feature vector of a possibly ambiguous noun accordingly.

6 Conclusion & Future Work

In this paper, an extension of NMF has been presented that combines both bag of words data and syntactic data in order to find latent semantic dimensions according to which both words and syntactic relations can be classified. The use of three way data allows one to determine which dimension(s) are responsible for a certain sense of a word, and adapt the corresponding feature vector accordingly, 'subtracting' one sense to discover another one. When embedded in a clustering framework, the method provides a fully automatic way to discriminate the various senses of words. The evaluation against EuroWordNet shows that the algorithm is genuinely able to disambiguate the features of a given word, and accordingly its word senses.

We conclude with some issues for future work. First of all, we would like to test the method that has been explored in this paper with other evaluation frameworks. We already mentioned the focus on ambiguous nouns, and the use of the new Cornetto database for Dutch. Next, we would like to work out a proper probabilistic framework for the 'subtraction' of dimensions. At this moment, the subtraction (using a cut-off) is somewhat ad hoc. A probabilistic modeling of this intuition might lead to an improvement.

And finally, we would like to use the results of our method to learn selectional preferences. Our method is able to discriminate the syntactic features that are linked to a particular word sense. If we can use the results to improve a parser's performance, this will also provide an external evaluation of the algorithm.

References

Church, Kenneth Ward and Patrick Hanks. 1990. Word association norms, mutual information & lexicography. *Computational Linguistics*, 16(1):22–29.

Harris, Z. 1985. Distributional structure. In Katz, Jerrold J., editor, *The Philosophy of Linguistics*, pages 26–47. Oxford University Press.

Hofmann, Thomas. 1999. Probabilistic latent semantic analysis. In *Proc. of Uncertainty in Artificial Intelligence, UAI'99*, Stockholm.

Landauer, Thomas and Se Dumais. 1997. A solution to Plato's problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. *Psychology Review*, 104:211–240.

Lee, Daniel D. and H. Sebastian Seung. 2000. Algorithms for non-negative matrix factorization. In *NIPS*, pages 556–562.

Lin, D. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of* COLING/ACL *98*, Montreal, Canada.

Manning, Christopher and Hinrich Schütze. 2000. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, Massachussets.

Ordelman, R.J.F. 2002. Twente Nieuws Corpus (TwNC), August. Parlevink Language Techonology Group. University of Twente.

Pantel, Patrick and Dekang Lin. 2002. Discovering word senses from text. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 613–619, New York, NY, USA. ACM Special Interest Group on Knowledge Discovery in Data, ACM Press.

Resnik, Philip. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI*, pages 448–453.

Schütze, Hinrich. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123.

van der Plas, Lonneke and Gosse Bouma. 2005. Syntactic contexts for finding semantically similar words. In van der Wouden, Ton et al., editors, *Computational Linguistics in the Netherlands 2004. Selected Papers from the Fifteenth CLIN Meeting*, pages 173–184, Utrecht. LOT.

van Noord, Gertjan. 2006. At Last Parsing Is Now Operational. In Mertens, Piet, Cedrick Fairon, Anne Dister, and Patrick Watrin, editors, *TALN06. Verbum Ex Machina. Actes de la 13e conference sur le traitement automatique des langues naturelles*, pages 20–42, Leuven.

Vossen, Piek et al. 1999. The Dutch Wordnet, July. University of Amsterdam.

Wu, Zhibiao and Martha Palmer. 1994. Verb semantics and lexical selection. In *32nd. Annual Meeting of the Association for Computational Linguistics*, pages 133–138, New Mexico State University, Las Cruces, New Mexico.