A Simple Extension of Non-negative Matrix Factorization for Three Way Data

Tim Van de Cruys

Humanities Computing, University of Groningen
Oude Kijk in 't Jatstraat 26, 9712 EK Groningen, The Netherlands
t.van.de.cruys@rug.nl,
WWW home page: http://www.let.rug.nl/decruys/

Abstract. Factorization algorithms are able to represent a large amount of data in a succinct way. Most factorization algorithms deal with two-way data, but often, one wants to analyze data that is interconnected in more than two ways. A number of methods have been developed to analyze three-way data, but those are generally not capable of handling massive data sets. In this paper, an extension of non-negative matrix factorization is presented that is able to deal with three-way data in an efficient way. The algorithm calculates the pairwise co-occurrence data for each mode separately – making it computationally efficient – while retaining the three-way structure that is present in the data. The approach is applied to the problem of word sense discrimination, for which the method reaches state-of-the-art performance.

1 Introduction

Factorization algorithms have since long been a popular research topic in the machine learning community, mainly because of two reasons. First of all, a factorization is able to express an abundance of (overlapping) features in terms of a limited number of components, making it easier to handle the data computationally. Secondly, factorization algorithms are generally able to overcome data sparseness and noise, yielding a representation of the data that goes beyond the initial results. These two characteristics make them an attractive tool for machine learning algorithms dealing with massive data sets. Consequently, they have been used in various domains, ranging from image recognition over gene expression analysis to information retrieval.

Most factorization algorithms deal with two-way data, taking a two-dimensional array – a matrix – as their input. There are, however, many situations in which one would like to analyze data that is represented in more than two modes. In image recognition, for example, one might want to analyze images in various lighting conditions; in information retrieval, one might want to model users by queries by documents. A number of methods have been developed to analyze three-way data sets, such as THREE WAY PRINCIPAL COMPONENT ANALYSIS and more recently NON-NEGATIVE TENSOR FACTORIZATION. These methods

deal with data represented as three-dimensional arrays, called *tensors*. A problem with these methods, however, is that the computations will become difficult – if not impossible – with massive datasets. When gigabytes of data are taken into account – as is often the case with text mining applications – the tensor will grow too large, and computations will become infeasible, even with sparse implementations of the algorithms. In this paper, a method is described that is able to efficiently deal with massive three-way data sets, by looking at the pairwise co-occurrence information for each mode separately. Describing the method, we will mainly focus on text mining applications, but we believe the method can easily be applied in other domains where massive data sets make the application of genuine three-way methods impossible.

2 Previous Work

2.1 Two-way Factorizations

As mentioned above, most factorization algorithms focus on two-way data. One of the best known algorithms is PRINCIPAL COMPONENT ANALYSIS (PCA) [1]. PCA involves an eigenvalue decomposition of a data covariance matrix. The original data is transformed into a new coordinate system, in which the eigenvectors with the heighest eigenvalues indicate the dimensions with the greatest variance. Keeping a limited number of dimensions, the transform yields the best possible fit in a least square sense. SINGULAR VALUE DECOMPOSITION (SVD) is the generalization of the eigenvalue decomposition used in PCA. [2]

In information retrieval, singular value decomposition has been applied in LATENT SEMANTIC ANALYSIS (LSA [3, 4]). 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 SVD. The most important dimensions that come out of the SVD allegedly represent 'latent semantic dimensions', according to which nouns and documents can be represented more efficiently.

LSA has been criticized for not being the most appropriate data reduction method for certain applications such as text mining. 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 [5, §15.4.3]. Successive methods such as PROBABILISTIC LATENT SEMANTIC ANALYSIS (PLSA) [6], 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.

2.2 Three-way Factorizations

To be able to cope with three-way data, several algorithms have been developed. In statistics, three-way component analysis has been extensively investigated

(for an overview, see [7]). The two most popular methods are PARALLEL FACTOR ANALYSIS (PARAFAC) [8, 9] and THREE-MODE PRINCIPAL COMPONENT ANALYSIS (3MPCA) [10]. Singular value decomposition has been generalized for multi-way data as HIGHER ORDER SINGULAR VALUE DECOMPOSITION (HOSVD) [11], and applied in various domains such as image recognition [12]. One last important method dealing with multi-way data is the generalization of non-negative matrix factorization, called NON-NEGATIVE TENSOR FACTORIZATION [13].

A problem with these methods is that the computations will be become difficult – if not impossible – as the data sets get larger.

3 Algorithm

3.1 Non-negative Matrix Factorization

Non-negative matrix factorization (NMF) [14] 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. Convergence of the update rules to a local optimum is proven in [14].

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

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

3.2 Extending Non-negative Matrix Factorization

We now propose an extension of NMF that is able to cope with multi-way data in a computationally efficient way. A factorization with three modes is taken as example, but the method can easily be extended to more modes.

As we are interested in finding latent structure among three-way data, but representing the data in a three-way array is computationally infeasible (as is the case with massive data sets), we proceed from the pairwise co-occurrence matrices for each of the modes. Thus, we construct three matrices. The first matrix contains mode 1 cross-classified by mode 2, the second matrix contains mode 2 cross-classified by mode 3, and the third matrix contains mode 1 cross-classified by mode 3. Admittedly, aggregating over the three-way data means that we throw away a lot of information, but this allows us to represent three-way data in a concise manner, and – as will be shown – the latent three-way structure will still be retrievable.

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

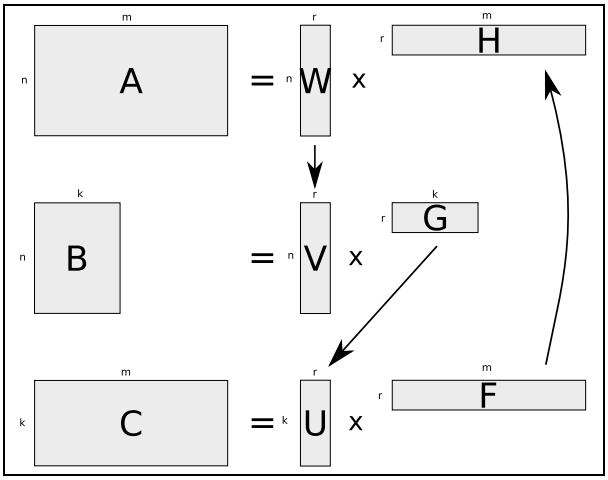

Fig. 1. A graphical representation of the extended NMF

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.

The set-up of the algorithm leaves some room for variation, depending on which matrix we initialize at random, and which matrices we calculate by using the update rules. These variations do not seem to influence the algorithm vitally.

4 Application: Word Sense Discrimination

4.1 Introduction

Many words used in natural language are ambiguous: they have various senses. Traditional algorithms dealing with semantic similarity cannot cope with this ambiguity.[1] By using the extended NMF algorithm presented in 3.2, both 'bag of words' data and syntactic data can be classified according to topical dimensions. 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 topical dimensions found by the bag of words approach.

4.2 Basic NMF

NMF can be straightforwardly applied to create semantic word models. In the example underneath, 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 (1). 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.

(1) 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'

4.3 Extended NMF

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

[1] By traditional algorithms, we mean algorithms that calculate the semantic similarity of words according to their distributional similarity – the context in which they appear. Vectors are created for each word, and the similarity is calculated in vector space. For a detailed description, see [15].

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.[2]

In (2), 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).

(2) 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'

4.4 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:

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

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 $\overrightarrow{h}_{dim}$.

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.

[2] Dependency relations of the noun *apple* might be 'object of *eat*' and 'adjective *red*'. Context words appearing in *apple*'s context window might be words like *tree* and *basket*.

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 4.4. 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.

(3) 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 (3) 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 (4).

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

b. *Barcelona*, *Milaan* 'Milan', *München* 'Munich', *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.

4.5 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 a standard K-means algorithm to create the initial cluster centroids. The initial vectors to be clustered are adapted with pointwise mutual information [16].

In (5), an example of our clustering algorithm with initial K-means clusters is given. 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) 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'

4.6 Evaluation

Methodology The clustering results are evaluated according to Dutch EuroWordNet [17]. 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 method is compared to Pantel and Lin's [18].

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 [19], Resnik's [20] and Lin's [21]. In this evaluation, Wu & Palmer's [19] measure will be adopted.

To calculate precision, we apply the same methodology as Pantel and Lin [18].[3] 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:

$$sim_W(s,u) = \max_{t \epsilon S(u)} sim(s,t) \tag{5}$$

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:

$$sim_C(s,c) = \frac{\sum_{u \epsilon c_k} simW(s,u)}{k} \tag{6}$$

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

$$\max_{s \epsilon S(w)} simC(s,c) > \theta \tag{7}$$

and the EuroWordNet sense of w that corresponds to c is

$$\arg\max_{s \epsilon S(w)} simC(s,c) \tag{8}$$

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.[4] Precision and recall of a clustering algorithm is the average precision and recall of all test words.

Experimental Design We have applied the interleaved NMF presented in section 3.2 to Dutch, using the TWENTE NIEUWS CORPUS [22], 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 [23], and dependency triples have been extracted. Next, the three matrices needed for our method have been constructed: one containing nouns by dependency relations (5K × 80K), one containing nouns by context words (5K × 2K) and one containing dependency relations by context words (80K × 2K). We did 200 iterations of

[3] Note, however, that our similarity measure is different. Where Pantel and Lin use Lin's [21] measure, we use Wu and Palmer's [19] measure.

[4] 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.

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.

Results Table 1 shows precision and recall figures for three different algorithms, according to various similarity thresholds θ (equation 7). kmeans$_{nmf}$ describes the results of our algorithm with K-means clusters, as described in section 4.5. For comparison, we have also included the results of the original CBC algorithm (CBC$_{orig}$) as described by Pantel and Lin [18] – considered the state-of-the-art algorithm for word sense discrimination – and the results of a standard K-means clustering (kmeans$_{orig}$, $k = 600$), in which each word is only assigned to its predominant sense.

		threshold θ			
		.40 (%)	.50 (%)	.60 (%)	.70 (%)
kmeans$_{nmf}$	prec.	78.97	69.18	55.16	39.01
	rec.	63.90	55.95	44.77	31.72
CBC$_{orig}$	prec.	44.94	38.13	29.74	21.61
	rec.	69.61	60.00	48.00	35.87
kmeans$_{orig}$	prec.	86.13	74.99	58.97	41.54
	rec.	60.23	52.45	41.80	29.88

Table 1. Precision and recall for three different algorithms according to various 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.

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[5], 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. The three way data

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

factorization 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.

In order to compare the performance of the different algorithms, the F-measure has been calculated, which combines precision and recall. Table 2 shows two variants of the F-measure for similarity threshold $\theta = .50$. F_1 is the standard F-measure, giving equal weight to recall and precision. F_2 weighs recall twice as much as precision (reflecting the importance of finding the multiple senses of a word).

	F_1 (%)	F_2 (%)
kmeans_{nmf}	61.86	58.17
CBC_{orig}	46.63	53.82
kmeans_{orig}	61.73	55.80

Table 2. F-measures for three different algorithms (similarity threshold $\theta = .50$)

When precision and recall are weighed equally (F_1) kmeans_{nmf} and kmeans_{orig} score about the same, both outperforming CBC_{orig} by $\pm$ 15 %. But when we emphasize the induction of multiple word senses (F_2 - recall is twice as important as precision), kmeans_{nmf} clearly outperforms both other algorithms.

5 Conclusion

In this paper, an extension of NMF has been presented that is able to deal with three-way data in an efficient way. Whereas full-fledged three-way factorization algorithms run into computational problems for massive data sets, our algorithm – calculating the pairwise co-occurrence data for each mode separately – is computationally efficient, while retaining the three-way structure that is present in the data.

The approach has been applied to the problem of word sense discrimination, and the results indicate that the algorithm is able to capture the latent three-way structure present in the data; the combination of bag of words data and syntactic 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.

References

1. Pearson, K.: On lines and planes of closest fit to systems of points in space. Philosophical Magazine **2**(6) (1901) 559–572
2. Wall, M.E., Rechtsteiner, A., Rocha, L.M.: 5. In: Singular Value Decomposition and Principal Component Analysis. Kluwel, Norwell, MA (Mar 2003) 91–109
3. Landauer, T., Dumais, S.: A solution to Plato's problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. Psychology Review **104** (1997) 211–240
4. Landauer, T., Foltz, P., Laham, D.: An Introduction to Latent Semantic Analysis. Discourse Processes **25** (1998) 295–284
5. Manning, C., Schütze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge, Massachussets (2000)
6. Hofmann, T.: Probabilistic latent semantic analysis. In: Proc. of Uncertainty in Artificial Intelligence, UAI'99, Stockholm (1999)
7. Kiers, H., van Mechelen, I.: Three-way component analysis: Principles and illustrative application. Psychological Methods (6) (2001) 84–110
8. Harshman, R.: Foundations of the parafac procedure: models and conditions for an "explanatory" multi-mode factor analysis. In: UCLA Working Papers in Phonetics. Volume 16., Los Angeles, University of California (1970) 1–84
9. Carroll, J.D., Chang, J.J.: Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. Psychometrika (35) (1970) 283–319
10. Tucker, L.: Some mathematical notes on three-mode factor analysis. Psychometrika (31) (1966) 279–311
11. De Lathauwer, L., Moor, B.D., Vandewalle, J.: A multilinear singular value decomposition. SIAM Journal on Matrix Analysis and Applications **21**(4) (2000) 1253–1278
12. Vasilescu, M.A.O., Terzopoulos, D.: Multilinear analysis of image ensembles: Tensorfaces. In: ECCV. (2002) 447–460
13. Shashua, A., Hazan, T.: Non-negative tensor factorization with applications to statistics and computer vision. In: ICML '05: Proceedings of the 22nd international conference on Machine learning, New York, NY, USA, ACM (2005) 792–799
14. Lee, D.D., Seung, H.S.: Algorithms for non-negative matrix factorization. In: NIPS. (2000) 556–562
15. Van de Cruys, T.: Semantic clustering in dutch. In Sima'an, K., et al., eds.: Proceedings of the Sixteenth Computational Linguistics in the Netherlands (CLIN), University of Amsterdam 17–32
16. Church, K.W., Hanks, P.: Word association norms, mutual information & lexicography. Computational Linguistics **16**(1) (1990) 22–29
17. Vossen, P., et al.: Eurowordnet, building a multilingual database with wordnets for several european languages website.
18. Pantel, P., Lin, D.: Discovering word senses from text. In: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, New York, NY, USA, ACM Special Interest Group on Knowledge Discovery in Data, ACM Press (2002) 613–619
19. Wu, Z., Palmer, M.: Verb semantics and lexical selection. In: 32nd. Annual Meeting of the Association for Computational Linguistics, New Mexico State University, Las Cruces, New Mexico (1994) 133–138

20. Resnik, P.: Using information content to evaluate semantic similarity in a taxonomy. In: IJCAI. (1995) 448–453
21. Lin, D.: Automatic retrieval and clustering of similar words. In: Proceedings of COLING/ACL 98, Montreal, Canada (1998)
22. Ordelman, R.: Twente Nieuws Corpus (TwNC) (August 2002) Parlevink Language Technology Group. University of Twente.
23. van Noord, G.: At Last Parsing Is Now Operational. In Mertens, P., Fairon, C., Dister, A., Watrin, P., eds.: TALN06. Verbum Ex Machina. Actes de la 13e conference sur le traitement automatique des langues naturelles, Leuven (2006) 20–42