AALTO UNIVERSITY
School of Electrical Engineering

Jyri Soppela

NONNEGATIVE MATRIX FACTORIZATION IN TEXT MINING APPLICATIONS

Thesis submitted for examination for the degree of Master of Science in Technology

Espoo 23.09.2014

Thesis supervisor:

Prof. Erkki Oja

Thesis instructor:

D.Sc. Ricardo Vigário

Author: Jyri Soppela

Title: Nonnegative Matrix Factorization in Text Mining Applications

Date: 23.09.2014 Language: English Number of pages: 6+45

Faculty: School of Electrical Engineering

Professorship: Computer and Information Science Code: T-61

Supervisor: Prof. Erkki Oja

Instructor: D.Sc. Ricardo Vigário

Meta-analysis of scientific publications is a practice where conclusions, sometimes novel, are drawn from already published material. It is mostly done by hand but on some fields, automatic tools have appeared to mine through large amounts of scientific literature. In this thesis, methods in statistical processing of natural language are used to process neuroscience articles. The long-time goal in which this thesis is a part is to construct a method to automatically process neuroscience publications and possibly by combining data in them, find new results not found by the original authors. Two computational methods, k-means clustering and non-negative matrix factorization, were used on several text data datasets to find semantic structure in them. The results using the computational methods were not very useful but proved that the tf-idf feature extraction method has potential. The clustering performed better than random assignment of clusters and published literature has presented even higher results using the same methods with different parameters.

Keywords: nonnegative matrix factorization, text mining, tf-idf

Preface

I would like to thank Dr. Ricardo Vigario for instruction and Prof. Erkki Oja for supervision. Also, Id'd like to thank Mr. Nima Reyhani, Mr. Ilari Nieminen and Dr. Zhirong Yang for helping out when needed and adding valuable insight to the problems at hand.

Espoo, 23.09.2014

Jyri Soppela

Contents

Symbols and Abbreviations

$\mathbf{A}$	Matrix
$\mathbf{a}$	Vector
a	Scalar value

SNLP	Statistical natural language processing
NMF	Non-negative matrix factorization
PCA	Principal component analysis
PCFG	Probabilistic context-free grammar
PLSI	Probabilistic latent semantic indexing
SVD	Singular value decomposition
tf-idf	term frequency - inverse document frequency

1 Introduction

Recently a number of text mining tools to query the vast amounts of research data available in the fields of medical and biological studies have been introduced[2]. These text mining tools respond to the growing need to quickly analyze publications as number of studies and amount of data produced by experiments grow. Also, meta-analysis is needed to construct broad overviews by using the single publications as building blocks. Database and information mining have been succesful approaches, especially when a leading institution in the field has made efforts in building tools for querying its large databases.

In neuroscience, similar meta-analysis tools would be very useful, as the amount of available studies is large and complex neurological behavior is rarely visible in a single study. There has been previous text mining work on neuroinformatics data [10, 38], but currently there's no universal database in use.

One approach to create meta-analysis tools is to make article writers add meta-data to their entries. In many journals, the articles contain a set of keywords but little else. There have also been structured databases with the goal to allow queries to database fields, but there's still no universal neuroscience database in use.

Another approach, the one considered in this thesis, is automated text and image mining. This does not require a uniform database structure or human interference for adding articles to the databases. Automatic document mining can take advantage of the fact that scientific articles have a rather well-defined structure, regardless of the publisher. The section fields are standard and appear in standard order, which gives crucial information about the expected topics in each text section.

There has been previous study in the Aalto University (former Helsinki University of Technology) about automated neuroscience meta-analysis in the field of image analysis. The large-scale goal is to construct a hierarchical analysis tool that takes into account the text mining results, image analysis results and the structure of the article and helps the user to search for relevant articles according to e.g. experimental methods, stimulus types or relevant anatomical structures in the brain.

A hierarchically structured meta-analysis tool can also evaluate the congruence of the results and make an estimate of the certainty of the analysis. This is neccessary and useful, as automated text mining tools tend to have a fairly large error rate.

Automatic processing of natural languages is a field of machine learning that has been studied for decades using several different approaches. The problems vary from machine translations to text queries, and several different methods are used for modelling text and representing semantics in a machine processable format. Many of the problems encountered stem from the fact that language originating from human socities is not directly processable by computers.

As natural languages have defined grammar rules, it would be intuitive to start studying the grammatic structure of natural languages and then model the form in which information is passed in them. However, already in 1921 it became clear that

"All grammars leak" [47] and modeling language as a strictly rule-obeying process is impossible.

Currently a popular way to extract information from text documents is to compare the frequencies of different terms that appear in those documents. This seems reasonable as important terms should appear more often than irrelevant ones. However, even this simple task ends up being less simple when non-informative grammatic words like prepositions and articles have to be considered. In addition, comparing the relative frequencies of text documents produces very high-dimensional data that requires dimensionality reduction methods. Comparing the frequencies of terms while ignoring text structure is called the "bag of words" -approach.

For this application, a computational method called nonnegative matrix factorization (NMF) can be used to analyze the sparse and high-dimensional data. Previous applications of NMF on text mining data have been successful, as NMF has several advantages when applied to text data[12]. First of all, NMF works only on nonnegative data as its name implies. The bright side of this is that it also produces strictly nonnegative results, which makes interpretation easier in case of e.g. text data. The bag of words approach mentioned earlier produces term counts as features, which are never negative. Furthermore, the factorization matrices can be interpreted as topic components.

This thesis explains the NMF and several algorithms that optimize the factorization. In addition, the Reuters benchmark data set is presented as well as a scientific article dataset collected from the Neuroimage journal releases. Both of these data sets are mined for semantic content using NMF and k-means, a standard clustering algorithm.

The next chapter will explain the basic concepts of statistical natural language processing without going into detail about the mathematical methods used.. The third chapter will explain the algorithms and methods used in this thesis, as well as additional, related methods. The fourth chapter will present results to the Reuters and Neuroimage dataset clustering problems. The fifth and final chapter includes analysis of the results and presents possible lines for future research.

2 Statistical natural language processing

The Turing test proposed by Alan Turing [44] might well be the most famous challenge for natural language processing and machine learning. In the Turing test a human judge has a conversation with a machine that generates its answers using natural language processing and other fields of machine learning. In case the human judge cannot determine whether the other conversant is a human or machine, the machine is considered to have passed the Turing test.

Since 1950, when the test was proposed, natural language processing problems have been approached from several angles. Earlier approaches focused on modeling the semantic meanings in language-neutral ways similar to Noam Chomsky's *transformational grammar*. [8] The basic assumption of this approach is that the deep semantic structures in natural languages are similar in all languages. These deep structures could be used to translate languages to another and also to construct a language-neutral semantic syntax.

In contrast to logical syntax models, statistical natural language processing (SNLP) handles the natural languages using stochastic and statistical approaches. Instead of presenting grammar as a strict logical process, SNLP models it as a stochastic process. [37]

2.1 Applications

In this study, statistical natural language processing is used to classify neuroscience publications. The application type is thus document classification. This is by no means the only case where it is useful to apply statistical methods on natural language in written text form. The methods explained in this section can also be used in e.g. following cases.

2.1.1 Machine translation

As mentioned above in the section 1, attempts to model information in natural language in exact form have not been successful. When translating semantic meanings from one language to another, additional problems arise as meanings do not neccessarily map exactly from one language to another. Stochastic approach in automatic translations is thus a viable approach. Even in this case, there are limits, as is noted by Madsen. [35]

2.1.2 Information retrieval

Information retrieval is the field of finding relevant information sources from a collection of information resources. Internet search engines such as Google are a good example of this. Automatic information retrieval has grown to be an extremely im-

portant field, as it has grown to be often the preferred method to gain information, even surpassing asking other people. [36]

2.1.3 Document classification

The subject of this thesis is document classification. In this field, each document is given one or more classifications or categories. In reality, documents rarely follow strict category limits. However, document classification using popular classification methods has its merits. Results gained from classification methods are easy to compare and there is plenty of research literature about implementing different classification methods on standardized document corpuses.

2.1.4 RNA structure modeling

Surprisingly, the concept of stochastic grammar has been used in computational biology to model RNA structures [46]. As the term would suggest, modeling RNA structure as generated by a probabilistic context-free grammar (PCFG) is not conceptually different from modeling natural languages as emerging from and underlying PCFG.

2.2 Corpus linguistics

A method of language analysis called corpus linguistics approaches the language solely from the angle of how it is used. Corpus linguistics does not introduce a priori knowledge of how the language should be used, what is its correct grammar and what is the acceptable vocabulary. [5, 49]

A purely corpus linguistic approach is language-neutral, as the analysis is derived solely from the source material. In this thesis a pure corpus linguistic approach is not used, but instead language pre-processing methods are used that assume the analyzed language to be English.

A commonly method used in corpus linguistics is annotation. Especially part-of-speech tagging is used for lexical analysis. The linguistic model used in this thesis is the vector space model, which does not require part-of-speech tagging and thus annotation is not used.

2.3 Vector space model

The text representation methods applied in this study use the “bag of words” approach that ignores the sentence structure and the order of words in a document. Instead, they focus on the relative frequency of different terms. A term is defined as a single word or a collocation of several words that always appear in the same order. This approach means that the text data can be fed to the system relatively

easily, as the model does not care for the word classes or structural meanings of the words. This makes it possible to handle text data without first adding syntax tags by hand or using advanced dictionaries. In addition, the bag of words -approach is not language-dependent, even though it would perform better with languages with as few morphological forms per term as possible. For example, the model fares well with English, but would have trouble with texts written in Finnish.

Each document is represented as a feature vector, where each element of the vector represents the numerical weight of one term in the document text. The document is then projected into a vector space that has the total of key terms found from the documents as its dimensions. The resulting data will be high-dimensional and sparse, as the total number of different terms in the text collection is likely to be very high and each document will contain only a small fraction of all the terms found in the data.

Both the high number of dimensions and the sparsity will present constraints to which numerical analysis tools are suitable. It is likely that many of the dimensions are mostly zero, and metrics such as Euclidean distance might not be very descriptive of the data.[37]

2.4 Tokenization

Tokenization is a process where the text is divided into tokens that each have a separate semantic meaning. Simply treating all words as tokens would be the simplest method, but this approach would include great amounts of noise into the features.

The division of text into words is done simply by splitting the raw text to words at each whitespace or newline character. To avoid noise, some expressions have to be filtered out. First, numerical expressions such as dates and measures are removed by accepting only tokens with alphabetic characters. In addition, very common words without significant semantic meaning such as “to” or “and” will be removed. Words without semantic content are listed in a stopword list specific to each language and then removed from the data. [37]

2.5 Stemming

As semantic information is not relevant for the bag of words approach, morphological information in words can be ignored. The process of unifying different variations of the same word with different morphological forms is called stemming. For example, “shoe” and “shoes” have the same word stem and thus the same semantic information we are interested in.

The data handled in the experiments is in English, so the words have relatively few morphological variations. Some examples of the same word gaining several different forms still exist. The most important of these are the plurals of nouns and personas of verbs.

The stemming algorithm used in this study is the Porter stemming algorithm [43] found in the Snowball package of R scripting language. The algorithm is explained below.

2.5.1 Porter Stemming Algorithm

Vowels is a group of characters that is defined as: A, E, I, O, U or Y preceded by a consonant. All the other characters are consonants.

Any set of consecutive consonants will be denoted as C. A set of consecutive vowels will be denoted as V. Thus, as every character is either a consonant or vowel a word always consists of alternating blocks of either vowels or consonants. Any word can be expressed with one of the four types:

CVCV...C
CVCV...V
VCVC...C
VCVC...V

All four can be represented as

$$[C]VCVC...[V]$$

and denoting m as an arbitrary natural number that will be called the measure of the word:

$$[C]\{VC\}^{m}[V]$$

The stemming is done by recognising and modifying the suffix of a word. This is done by a set of rules following the format:

$$\text{(condition) S1} \rightarrow \text{S2}$$

The format will replace suffix S1 with suffix S2 in case the word stem before S1 satisfies the given condition. Thus, for example

```
(m > 1) EMENT -> E
```

would transform “measurement” and “replacement” to “measure” and “replace” as they can both be represented as

$$C\{VC\}^{4}$$

as the condition for replacement is $m > 1$, the ending suffix can be replaced.

The conditions given may be for example that the stem contains vowels, the stem ends in an S and so on. These are denoted as:

```
*S - the stem ends with S (similar for other letters, e.g. *T for T)
*v*- the stem contains a vowel
*d - the stem ends with a double consonants (e.g. -TT, -SS)
*o - the stem ends cvc, where the second c is not W, X or Y (e.g. _WIL, -HOP)
```

Stemming conditions may also contain logical operators such as *and*, *or* and *not*:

```
m>1 and (*S or *T)
```

Several sets of conditions can be given to a single stemming rule. In this case, the longest matching condition will be chosen for the suffix replacement.

For the English language, the complete Porter stemming algorithm consists of five steps, each with a set of stemming rules.

Step 1a

```
SSES    -> SS          caresses-> caress
IES     -> I           ponies  -> poni
                       ties    -> ti
SS      -> SS          caress  -> caress
S       ->             cats    -> cat
```

Step 1b

```
(m>0) EED  -> EE     feed      -> feed
                     agreed    -> agree
(*v*) ED   ->        plastered -> plaster
                     bled      -> bled
(*v*) ING    ->        motoring  -> motor
                     sing      -> sing
```

It should be noted that "feed" does not stem into "f" as it is a $m = 0$ word, also known as *null word*.

If the second or third of the rules in Step 1b is successful, the following is done:

```
AT                              -> ATE           conflat(ed)    -> conflate
BL                              -> BLE           troubl(ed)     -> trouble
IZ                              -> IZE           siz(ed)        -> size
(*d and not (*L or *S or *Z))   -> single letter hopp(ing)      -> hopp
                                                 tann(ed)       -> tan
                                                 fall(ing)      -> fall
                                                 hiss(ing)      -> hiss
                                                 fizz(ed)       -> fizz
(m=1 and *o)                    -> E             fail(ing)      -> fail
                                                 fil(ing)       -> file
```

Step 1c

```
(*v*) Y  -> I         happy -> happi
                      sky   -> sky
```

Step 2

```
(m>0) ATIONAL   -> ATE  relational      -> relate
(m>0) TIONAL    -> TION conditional     -> condition
                        rational        -> rational
(m>0) ENCI      -> ENCE valenci         -> valence
(m>0) ANCI      -> ANCE hesitanci       -> hesitance
(m>0) IZER      -> IZE  digitizer       -> digitize
(m>0) ABLI      -> ABLE conformabli     -> conformable
(m>0) ALLI      -> AL   radicalli       -> radical
(m>0) ENTLI     -> ENT  differentli     -> different
(m>0) ELI       -> E    vileli          -> vile
(m>0) OUSLI     -> OUS  analogousli     -> analogous
(m>0) IZATION   -> IZE  vietnamization  -> vietnamize
(m>0) ATION     -> ATE  predication     -> predicate
(m>0) ATOR      -> ATE  operator        -> operate
(m>0) ALISM     -> AL   feudalism       -> feudal
(m>0) IVENESS   -> IVE  decisiveness    -> decisive
(m>0) FULNESS   -> FUL  hopefulness     -> hopeful
(m>0) OUSNESS   -> OUS  callousness     -> callous
(m>0) ALITI     -> AL   formaliti       -> formal
(m>0) IVITI     -> IVE  sensitiviti     -> sensitive
(m>0) BILITI    -> BLE  sensibiliti     -> sensible
```

Step 3

```
(m>0) ICATE -> IC       triplicate  -> triplic
(m>0) ATIVE ->          formative   -> form
(m>0) ALIZE -> AL       formalize   -> formal
(m>0) ICITI -> IC       electriciti -> electric
(m>0) ICAL  -> IC       electrical  -> electric
(m>0) FUL   ->          hopeful     -> hope
(m>0) NESS  ->          goodness    -> good
```

Step 4

```
(m>1) AL                ->          revival     -> reviv
(m>1) ANCE              ->          allowance   -> allow
(m>1) ENCE              ->          inference   -> infer
```

```
(m>1) ER                    ->          airliner    -> airlin
(m>1) IC                    ->          gyroscopic  -> gyroscop
(m>1) ABLE                  ->          adjustable  -> adjust
(m>1) IBLE                  ->          defensible  -> defens
(m>1) ANT                   ->          irritant    -> irrit
(m>1) EMENT                 ->          replacement -> replac
(m>1) MENT                  ->          adjustment  -> adjust
(m>1) ENT                   ->          dependent   -> depend
(m>1 and (*S or *T)) ION    ->          adoption    -> adopt
(m>1) OU                    ->          homologou   -> homolog
(m>1) ISM                   ->          communism   -> commun
(m>1) ATE                   ->          activate    -> activ
(m>1) ITI                   ->          angulariti  -> angular
(m>1) OUS                   ->          homologous  -> homolog
(m>1) IVE                   ->          effective   -> effect
(m>1) IZE                   ->          bowdlerize  -> bowdler
```

Step 5a

```
(m>1) E ->                              probate     -> probat
                                        rate        -> rate
(m=1 and not *o) E          ->          cease       -> ceas
```

Step 5b

```
(m > 1 and *d and *L)    -> single letter   controll -> control
                                            roll     -> roll
```

The algorithm was first presented by M.F. Porter in 1980 [43].

2.6 N-grams

In addition to counting singular words, combinations of N consecutive words called N-grams can also be counted as analysable terms. Thus, 1-grams would represent single-word terms, 2-grams represent word pairs, *et cetera*. Using N-grams helps finding combination terms that have a separate meaning when combined than when separated. For example, neither of the words "spinal" or "cord" have the same meaning as "spinal cord".

The downside of taking N-grams into account is that they increase greatly the dimensionality of the vectors in vector space model. In the worst theoretical case, a document corpus with k different terms would have its vector space dimensionality increased from k to k^N when extracting N-grams with N consecutive terms. In practice many words never appear in conjuction to each other, but the data dimensionality is still significantly increased.

In the experiments performed in this thesis, the N-gram model is not used but it is introduced for future research and to give background to statistical processing of natural language. The decision not to use the N-gram model but to stay in one term units is due to computational limitations.

2.7 Tf-idf

Term frequency - inverse document frequency (tf-idf) is a numerical weight given to a term, according to its frequency in a document and its frequency in the data in general. It is used in this study to construct a feature vector where each item in the vector corresponds to the numerical weight of one term [37]. The tf-idf feature of a block of text allows for the use of vector space model and numerical comparison of different units of language.

Tf-idf is the product of term frequency ($\mathrm{tf}_{i,j} = \frac{n_{i,j}}{\Sigma_k n_{k,j}}$) and inverse document frequence ($\mathrm{idf}_i = \log \frac{|D|}{|\{d:t_i \in d\}|}$). Term frequency for term i in document j is calculated by dividing the number of its occurrences in a given document $n_{i,j}$ with the total amount of terms in document j.

Term frequency tells the importance of a term in one document, and is independent from the length of the document. Thus terms in a longer document will not gain higher weights, if the relative frequency does not change.

Inverse document frequency of term i tells how common the term is in general. $|D|$ is the total number of documents in the material and $|\{d : t_i \in d\}|$ is the number of documents that include term i. Using the inverse document frequency, it is possible to deal with terms that are very common although not carrying any information about the document. For example, most functional words like "the" or "in" appear in every document and thus are regarded as non-informative, as for them $\mathrm{idf}_i = 0$.

2.8 Latent Semantic Indexing

The probability of term appearance in documents is not random. Two terms with similar semantic meaning have a larger probability to appear in the same document than two terms that are completely unrelated. This information can be used to derive semantic connections between words. Latent Semantic Indexing is a method where the feature vector containing weights for each term is projected onto a lower-dimensional space, where the dimensions describe latent semantic content instead of term weights. [7, 37]

LSI is a dimensionality reduction method, because the dimensionality of the term space is larger than the dimensionality of the latent semantic space. Easiest way to derive the LSI space is to use PCA (Principal Component Analysis), which projects the term vectors onto a lower-dimensional orthogonal space, which maximally preserves the variance of the original data. LSI assumes that the semantic components are orthogonal, which may be a false assumption in the case of overlapping semantic

components.

3 Algorithms and Methods

3.1 Singular Value Decomposition

Singular value decomposition (SVD) is a linear algebra method that produces a factorization of matrix $\mathbf{X}$ by using its eigenvalues and eigenvectors. For the purposes of this thesis, matrix $\mathbf{X}$ is assumed to be real, but this requirement is not necessary in other applications as SVD is also possible for complex matrices.

For real matrices, singular value decomposition takes the format

$$\mathbf{X} \approx \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$
$$\begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mn} \end{pmatrix} = \begin{pmatrix} u_{11} & \cdots & u_{1m} \\ \vdots & \ddots & \vdots \\ u_{m1} & \cdots & u_{mm} \end{pmatrix} \begin{pmatrix} \Sigma_1 & \cdots & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \Sigma_r & 0 \end{pmatrix} \begin{pmatrix} v_{11} & \cdots & v_{n1} \\ \vdots & \ddots & \vdots \\ v_{1r} & \cdots & v_{nr} \end{pmatrix} \tag{1}$$

where the columns of $\mathbf{U}$ are orthonormalized eigenvectors of $\mathbf{X}\mathbf{X}^T$ and the columns of $\mathbf{V}$ are orthonormalized eigenvectors of $\mathbf{X}^T\mathbf{X}$, corresponding to the r largest eigenvectors. The matrix $\mathbf{\Sigma}$ contains the square roots of r first eigenvalues from $\mathbf{M}$ in descending order on its diagonal. The number r must be chosen suitably and according to the application at hand.

There are several mutually compatible interpretations of SVD: Singular value decomposition finds correlation in a data set and produces a mapping that sets the correlation of the data set into zero. Another interpretation is that the SVD finds the dimensions of the data that include the most variation. Finally, the SVD produces a mapping that gives the best possible linear approximation of the data when using less dimensions than in the original.

For the purposes of statistical natural language processing, SVD offers a method to significantly reduce data dimensionality without losing great amounts of variance in the data. SVD is also a useful tool to help the visualization of high-dimensional data. [22]

3.2 Principal Component Analysis

Principal component analysis (PCA) is a data processing method closely related to SVD. PCA can be used to reduce data dimensionality and to reduce noise. For example, it can be used to visualize high-dimensionality data in two dimensions with minimal loss of variance. PCA projects a data set into a space defined by the principal components of the data. This projection space has an equal or lower number of dimensions than the original data space. [29]

The first step of PCA is to normalize the data. The mean of the used data set $\mathbf{X}$ is subtracted from it, thus getting

$$\mathbf{X}_{norm} = \mathbf{X} - \overline{\mathbf{X}} \tag{2}$$

Then, the covariance matrix of the normalized data is calculated:

$$cov(\mathbf{X}_{norm}) = \mathbf{X}_{norm}\mathbf{X}_{norm}^T \tag{3}$$

Similarly as in SVD explained in chapter 3.1, the next step is to calculate the eigenvalues and unitary eigenvectors of the covariance matrix calculated in equation 3.

All the eigenvectors are orthogonal to each other, so together they define the same space as the dimensions of the original data. The eigenvector corresponding to the eigenvalue with the highest value is called the *principal component*, and corresponds to fitting a correlation line to the data. It is the vector that defines the direction in the data with most variance. Similarly, the eigenvector corresponding to the second largest eigenvalue corresponds to the dimension in the data that has most variance and is orthogonal to the principal component.

If there are strong correlations in the data, the eigenvectors corresponding to the smallest eigenvalues may well contain very litte variance. In this case, the original data can be projected into a space defined by only a part of the eigenvectors of the covariance matrix. Some of the variance will be lost, but the data dimensionality will be lower. In addition to compressing data, a projection into a lower-dimensional space often reduces noise.

To project the original data $\mathbf{X}$ into a lower-dimensional space, the selected eigenvectors are collected into a matrix that is called *feature vector*. The feature vector $\mathbf{F}$ is used to transform the original data into a new data set, where the dimensions have changed but where the relative differences of the data points and the structure of the data set are ideally the same.

The feature vector is thus

$$\mathbf{F} = \begin{pmatrix} \vec{eig}_1 & \vec{eig}_2 & \vec{eig}_3 & \cdots \end{pmatrix} \tag{4}$$

The projection $\mathbf{X}_{proj}$ is calculated as

$$\mathbf{X}_{proj} = \mathbf{F}^T \times \mathbf{X} \tag{5}$$

When calculating the projection, it should be noted that a transpose of the feature vector is taken and the eigenvectors are the rows of $\mathbf{F}^T$.

The dimensions of the projected data set do not directly correspond to the dimensions of the original data. Instead, the dimensions of the projected data are the eigenvectors derived from the covariance matrix and thus are not likely to be interpretable.

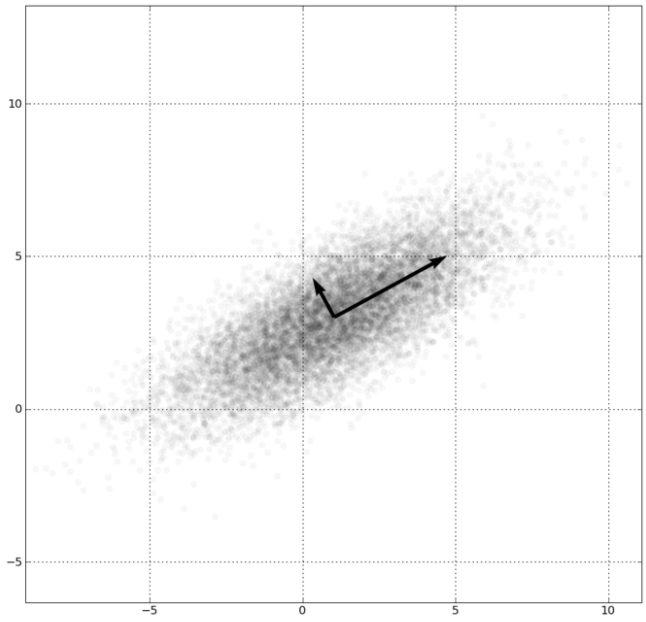

Figure 1: Two PCA-derived components of a two-dimensional data set. If both of the components derived from PCA are selected for reconstructing the data, the data can be reconstructed without loss. If only the first component is chosen, the data is projected along the component and data loss will occur.

In case PCA is used to compress data, the projected data must be returned to the coordinates of the original data. If only some of the eigenvectors of the covariance matrix were chosen for the projection, some information will be lost in the compression. Otherwise, the data will be completely the same as before projection.

The projected data is returned to the original coordinates by

$$\mathbf{X} = \left(\mathbf{F}^{-1}\right)^T \times \mathbf{X}_{proj} \tag{6}$$

3.3 k-means Clustering

K-means clustering is a classical method in machine learning for finding clusters in unlabeled or labeled data. The k-means method starts with the assumption that the data to be clustered contains observations from a Euclidean variable. Also, the clusters are thought of as groups of points whose internal variation is smaller than the variation between clusters. [6]

K-means represents each cluster in an N-dimensional dataset $\{\mathbf{x}_1 \dots \mathbf{x}_N\}$ with an M-dimensional vector. It does not estimate the number of clusters, but relies on a

given cluster number k. Thus, k-means defines a set of vectors $\{\boldsymbol{\mu}_k\}$, which act as *centroid* for the clusters. A data point belongs to a cluster, if its Euclidean distance to the cluster's centroid is smaller than the distance to any other centroid.

K-means doesn't provide an algorithm in itself, but rather an objective function that, when solved, will provide a clustering for the data. The objective function is the sum of the distances between each data point and the centroid of the cluster the data point belongs to. This objective function is also called *distortion measure*. The clustering tags the data points as binary vectors in matrix $\mathbf{r}_{nk}$, where each row $\mathbf{r}_n$ contains the cluster memberships of a data point $\mathbf{x}_n$ to the k clusters. As each data point belongs to just one cluster, each row of the membership matrix will have one value of 1 and the rest of the row will have only zeros. The k-means objective function is

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\mathbf{x}_n - \boldsymbol{\mu}_k||^2. \tag{7}$$

3.3.1 Algorithm

To find an optimum for the k-means objective function, a simple two-step algorithm can be applied. The centroids are initialized randomly. The first step of the algorithm, also called *assignment* step of the algorithm is to allocate each data point to one of the centroids:

$$R_i^{(t)} = \left[x_j \,\middle|\, ||\mathbf{x}_j - \boldsymbol{\mu}_i^{(t)}|| \leq ||\mathbf{x}_j - \boldsymbol{\mu}_l^{(t)}|| \text{ for all } l = 1, \ldots, K \right] \tag{8}$$

The second, or *update* step is to redefine the centroids as the means of each set of allocated data points:

$$\boldsymbol{\mu}_i^{(t+1)} = \frac{1}{\left| R_i^{(t)} \right|} \sum_{\mathbf{x}_j \in R_i^{(t)}} \mathbf{x}_j. \tag{9}$$

The algorithm repeats the assignment and update steps until two consecutive iteration rounds are identical. This means that between two iterations, no data points were re-assigned into another cluster. The stopping criterion is thus fulfilled if

$$\mathbf{R}^{(t)} = \mathbf{R}^{(t-1)} \tag{10}$$

An example of k-means clustering is provided in figures 2-5.

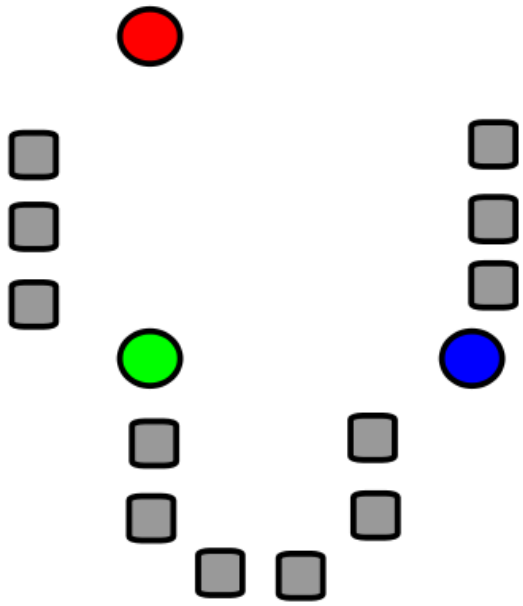

Figure 2: First step of the k-means clustering of a simple dataset. Random initialization of the clusters.

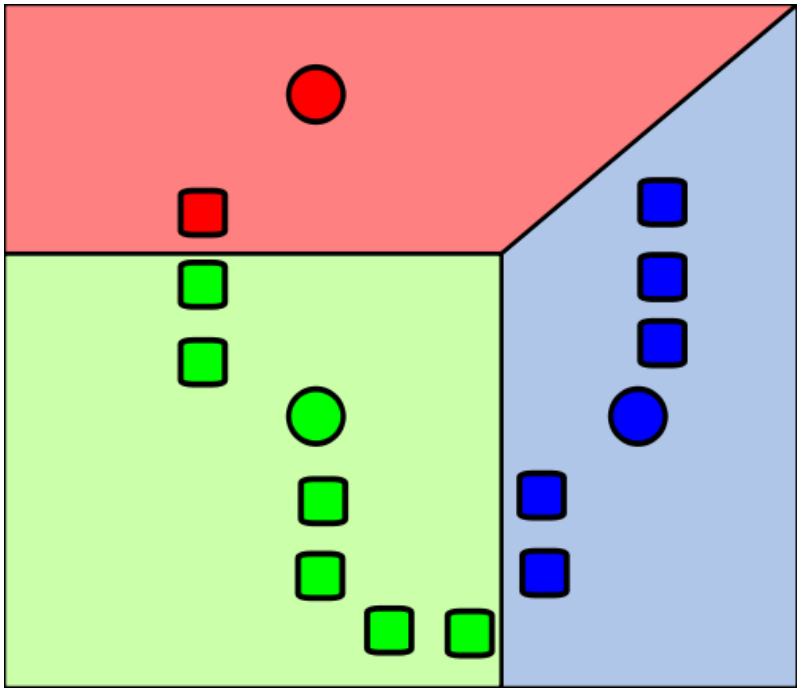

Figure 3: Second step of the k-means clustering of a simple dataset. Assignment of all data points to one cluster.

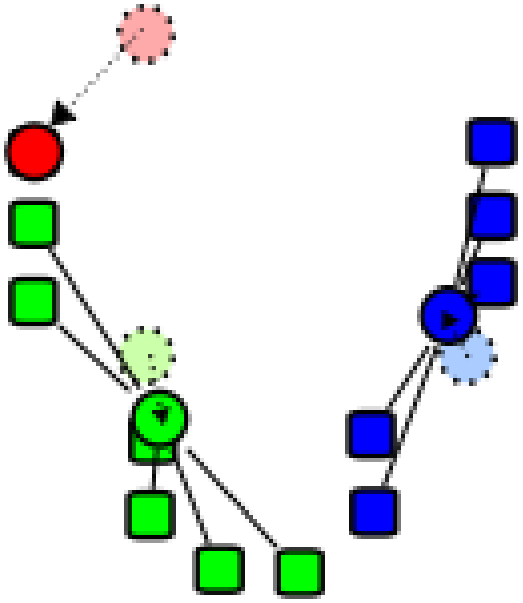

Figure 4: Third step of the k-means clustering of a simple dataset. Re-evaluation of the cluster centroids, based on the current clustering estimates.

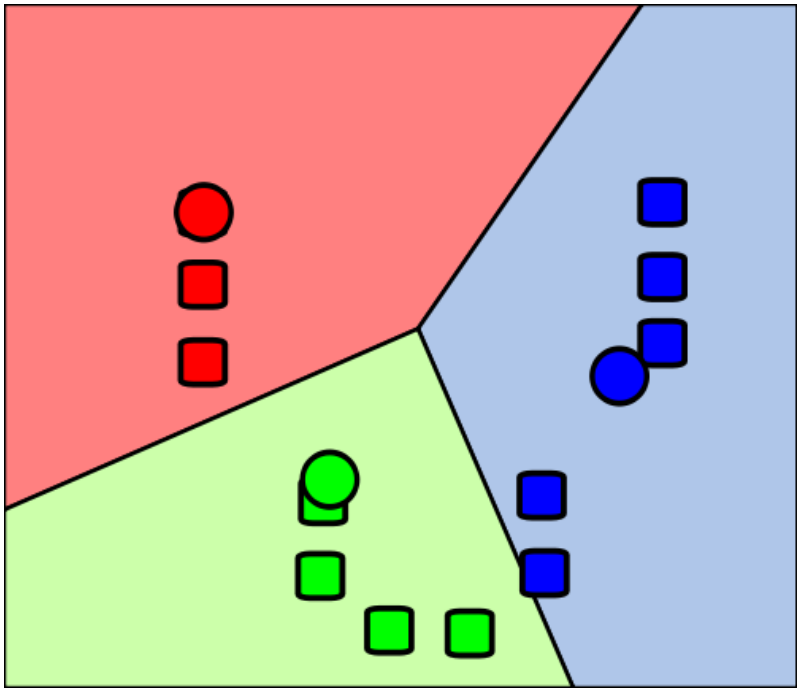

Figure 5: Fourth step of the k-means clustering of a simple dataset. Further iterations are performed until convergence is achieved and the stopping criterion is fulfilled.

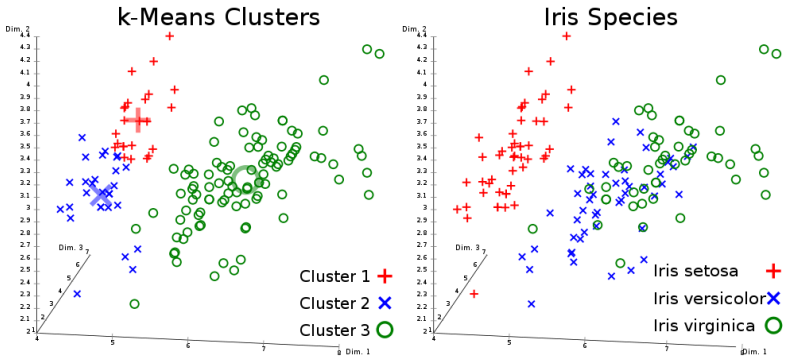

Figure 6: K-means clustering of the Iris dataset.The Iris species data set is a known benchmark data set where samples from three species of iris are measured [16]. The measured dimensions are sepal length, sepal width, petal length and petal width.

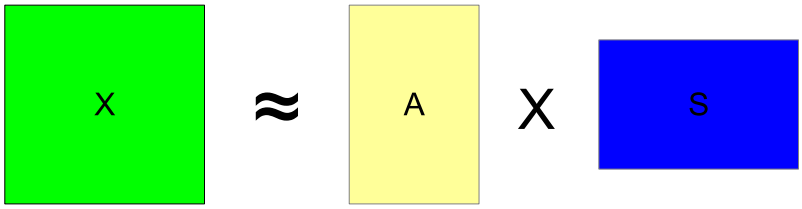

Figure 7: The basic nonnegative matrix factorization: The data matrix **X** is estimated as product of source matrices **A** and **S**.

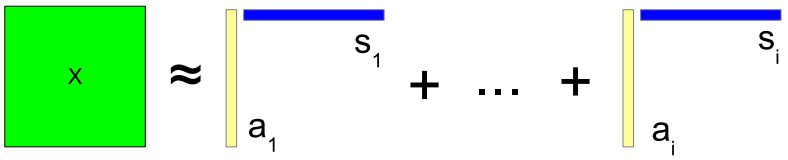

Figure 8: Row- and column-wise visalization of basic NMF.

3.4 Nonnegative Matrix Factorization

Nonnegative Matrix Factorization (NMF) is a dimension reduction method that requires used data to be nonnegative. Text document data represented in the vector space model mentioned in chapter 2.3 fulfills this criterion, as all features are based on term frequencies. NMF does not lose the nonnegativity attribute of the data during the dimension reduction, as for example principal component analysis (PCA) would. PCA is explained in chapter 3.2. [7]

NMF assumes that the observation vectors $\mathbf{x}_t$ can be estimated with a linear mixing model. Using this assumption the observations are composed of underlying source vectors, $\mathbf{s}_t$, that are mixed using matrix **A**: Thus the approximation of observation vector $\mathbf{x}_t$ using sources $\mathbf{s}_t$ and mixing vector **A** is

$$\mathbf{x}_t = \mathbf{A}\mathbf{s}_t + \boldsymbol{\epsilon}_t = \sum_n \mathbf{a}_n s_{nt} + \boldsymbol{\epsilon}_t. \tag{11}$$

Where ϵ_{nt} is the error term. When equation 11 is generalized into matrices, we get

$$\mathbf{X} \approx \mathbf{AS} \tag{12}$$

The dimension reduction by NMF is done by preselecting a number of sources p. If the size of the observation matrix **X** is $n \times m$, the size of the mixing matrix **A** will be $n \times p$ and the size of the source matrix **S** will be $p \times m$. To reduce the dimensionality of the data, p is chosen to be smaller than n.

The matrices **X**, **A** and **S** are required to contain only nonnegative elements. In cases when nonnegativity is a feature of the data, the nonnegativity requirement not only makes numerical computation simpler but also keeps the factorization truer to

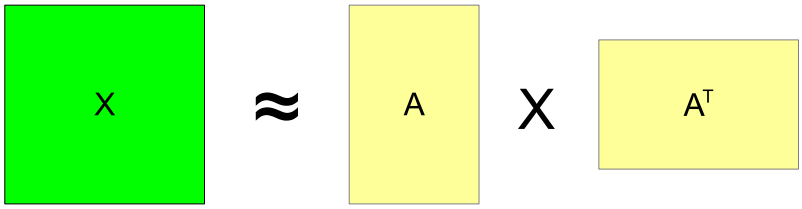

Figure 9: In symmetric NMF, there is only one source matrix $\mathbf{A}$, and the data matrix $\mathbf{X}$ is the product of $\mathbf{A}$ and its transpose $\mathbf{A}^T$.

the original data.

Approximating the matrices $\mathbf{A}$ and $\mathbf{S}$ is based on minimizing a distance function between the data $\mathbf{X}$ and its approximation $\mathbf{AS}$. In its general form, this optimization problem can be formulated as

$$\min_{\substack{\mathbf{A}\geq 0\\ \mathbf{S}\geq 0}} \mathcal{D}(\mathbf{X}; \mathbf{AS}) \tag{13}$$

The most commonly used distance measure J is the Euclidean distance, J_E, that is defined with Frobenius norm $||\mathbf{X}||_F$:

$$J_E(\mathbf{X}; \mathbf{AS}) = \frac{1}{2}||\mathbf{X} - \mathbf{AS}||_F^2. \tag{14}$$

Elementwise, the equation becomes

$$J_E(\mathbf{X}; \mathbf{AS}) = \frac{1}{2}\sum_{pt}\Big([\mathbf{x}]_{pt} - [\mathbf{AS}]_{pt}\Big)^2. \tag{15}$$

3.5 NMF variants

3.5.1 Symmetric NMF

The special case of NMF where $\mathbf{A} = \mathbf{S}$ is called symmetric NMF. Symmetric non-negative matrix factorization is thus given by

$$\mathbf{X} \approx \mathbf{A}\mathbf{A}^T \tag{16}$$

This model is considered equivalent to Kernel K-means clustering, as discussed further in section 3.10.1.

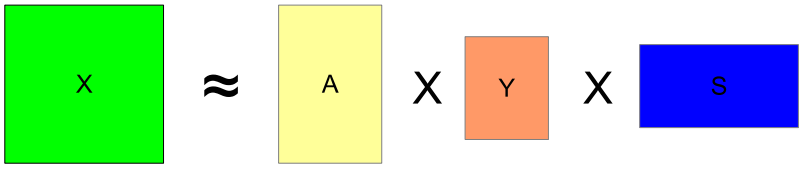

Figure 10: In tri-NMF, or three-factor NMF, the data matrix $\mathbf{X}$ is the product of three source matrices $\mathbf{A}$, $\mathbf{Y}$ and $\mathbf{S}$.

3.5.2 Semi-Orthogonal NMF

The Semi-Orthogonal NMF is subject to the same nonnegativity constraints as NMF, as well as to an orthogonality constraint

$$\mathbf{A}^T\mathbf{A} = \mathbf{I}, \text{ , or } \mathbf{S}\mathbf{S}^T = \mathbf{I} \tag{17}$$

The orthogonality constraint can be enforced by adding a transformation step to each iteration:

$$\mathbf{A} \leftarrow \left[\mathbf{A}^T\mathbf{A}\right]^{-1/2} \text{ , or } \mathbf{S} \leftarrow \left[\mathbf{S}\mathbf{S}^T\right]^{-1/2}\mathbf{S} \tag{18}$$

3.5.3 Tri-NMF

Tri-NMF, or three-factor NMF is a multi-layer NMF that constructs the factorization into three matrices with nonnegativity constraints:

$$\mathbf{X} \approx \mathbf{AYS} \tag{19}$$

In this factorization, $A \in \mathbb{R}^{I\times J}, \mathbf{Y} \in \mathbb{R}^{J\times K}, \mathbf{S} \in \mathbb{R}^{K\times L}$.

If no other constraints are introduced, the three-factor NMF is no different from the standard NMF [7]. The three-factor NMF is in this case reduced to standard NMF by $\mathbf{A} \leftarrow \mathbf{AY}$ or $\mathbf{S} \leftarrow \mathbf{YS}$. In case further constraints are introduced to the tri-NMF, the following variants of NMF differ from the standard NMF:

3.5.4 Orthogonal tri-NMF

Similar to semi-orthogonal NMF described in chapter 3.5.2, orthogonal tri-NMF introduces orthogonality constraints to the factorization matrices. In orthogonal tri-NMF, matrices $\mathbf{A}$ and $\mathbf{S}$ from tri-NMF equation 19 are orthogonally constrained such that

$$\mathbf{A}^T\mathbf{A} = \mathbf{I}, \text{ and } \mathbf{S}\mathbf{S}^T = \mathbf{I} \tag{20}$$

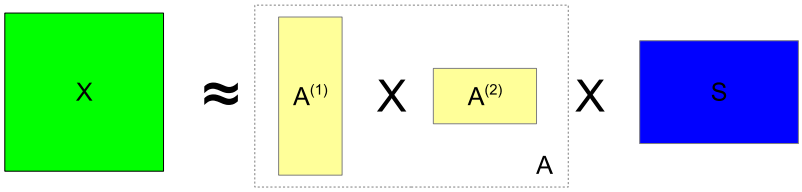

Figure 11: Multi-layer NMF is an iterative process, where the factor matrix $\mathbf{S}$ is further factorized in each iteration using any available NMF method. In this visualization, the multi-layer NMF is two iterations deep.

Matrix $\mathbf{Y}$ is an unconstrained matrix that can have both negative and nonnegative values.

3.5.5 Multi-layer NMF

Multi-layer NMF is a method of using several cascaded factorization matrices $\mathbf{A}^{(1)} \dots \mathbf{A}^{(N)}$ instead of single factorization matrix $\mathbf{A}$. Thus, the multi-layer NMF can be described as

$$\mathbf{X} \approx \mathbf{A}^{(1)}\mathbf{A}^{(2)} \dots \mathbf{A}^{(N)}\mathbf{S} \tag{21}$$

As a linear model, without further constraints all the matrices $\mathbf{A}^{(1)} \dots \mathbf{A}^{(N)}$ can be reduced into single matrix $\mathbf{A}$. Even without constraining the matrices $\mathbf{A}^{(1)} \dots \mathbf{A}^{(N)}$ further, the multi-layer NMF can be used as an iterative method for obtaining the final factorization matrix $\mathbf{A}$. This method demonstrably reduces conversion to local minima and improves NMF performance in cases of badly scaled or ill-conditioned data [14, 17, 18].

To gain the benefits of multi-layer NMF, the final factorization is gained by a hierarchical iteration. The first approximation is derived by $\mathbf{X} \approx \mathbf{A}^{(1)}\mathbf{S}^{(1)}$. Any appropiate NMF algorithm may be used. Then, matrix $\mathbf{S}^{(1)}$ is further factorized into component matrices: $\mathbf{S}^{(1)} \approx \mathbf{A}^{(2)}\mathbf{S}^{(2)}$. In separate iterations, different update rules and iteration parameters can be used. The factorization process can be continued as long as necessary or until predefined stopping criteria are fulfilled. The final result of the process is

$$\begin{aligned} \mathbf{X} &\approx \mathbf{A}^{(1)}\mathbf{A}^{(2)} \dots \mathbf{A}^{(N)}\mathbf{S}^{(N)} \\ \mathbf{A} &= \mathbf{A}^{(1)}\mathbf{A}^{(2)} \dots \mathbf{A}^{(N)} \\ \mathbf{S} &= \mathbf{S}^{(N)} \end{aligned} \tag{22}$$

3.5.6 Simultaneous NMF

Simultaneous NMF is a variant of NMF that deals with a problem where two different data matrices $\mathbf{X}_1$ and $\mathbf{X}_2$ are factorized so that one of the factorization matrices is shared. This is a problem that arises in for example gene expression research, where gene expression data and transcription factor regulation data are combined [7]

$$\begin{aligned} \mathbf{X}_1 &= \mathbf{A}_1\mathbf{S} \\ \mathbf{X}_2 &= \mathbf{A}_2\mathbf{S} \end{aligned} \tag{23}$$

3.6 Gradient Descent method for NMF

The typical blind source separation problem is the approximation of the factorization matrices $\mathbf{A}$ and $\mathbf{S}$, when only the observation matrix $\mathbf{X}$ is known. This is a problem with no single unique solution, and several methods exist to find a suitable factorization. A simple method for this is minimizing the Euclidean error measure in equation 14, using a gradient descent method. [7] The gradient of the error measure $\mathcal{J}_E$ is calculated, leading to the approximation updating the rule

$$\mathbf{S} \leftarrow \mathbf{S} - \eta \frac{\partial \mathcal{J}_E}{\partial \mathbf{S}}, \tag{24}$$

where η is an updating step coefficient. $\mathbf{S}$ is a matrix, so the partial derivative $[\partial \mathcal{J}_E / \partial \mathbf{S}]_{nt}$ can also be expressed as $\partial \mathcal{J}_E / \partial s_{nt}$. Thus, the update rule for individual terms is

$$s_{nt} \leftarrow s_{nt} - \eta_{nt} \frac{\partial \mathcal{J}_E}{\partial s_{nt}} \tag{25}$$

The update factor η_{nt} must be defined for all elements of $\mathbf{S}$, in other words for all combinations of (n, t).

To calculate the partial derivative, the cost function

$$J_E = \frac{1}{2}||\mathbf{X} - \mathbf{AS}||_F^2 = \frac{1}{2}\text{trace}((\mathbf{X} - \mathbf{AS})^{\mathrm{T}})(\mathbf{X} - \mathbf{AS})) \tag{26}$$

is considered to change for an infinitesimal amount ∂J_E as the matrix $\mathbf{S}$ to be approximated changes for an infinitesimal amount $\partial \mathbf{S}$. When J_E is differentiated with respect to the infinitesimal change $\partial \mathbf{S}$, the gradient of the error function can be formulated as shown in equation 27.

$$\frac{\partial J_E}{\partial s_{nt}} = \left[\mathbf{A}^{\mathrm{T}}\mathbf{X} - \mathbf{A}^{\mathrm{T}}\mathbf{AS}\right]_{nt} = -([\mathbf{A}^{\mathrm{T}}\mathbf{X}]_{nt} - [\mathbf{A}^{\mathrm{T}}\mathbf{AS}]_{nt}) \tag{27}$$

When equation 27 is substituted in equation 25, the element-wise update rule for gradient descent algorithm for matrix $\mathbf{S}$ becomes

$$s_{nt}s_{nt} \leftarrow s_{nt} + \eta_{nt}([\mathbf{A}^{\mathrm{T}}\mathbf{X}]_{nt} - [\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{S}]_{nt}) \tag{28}$$

In addition to the source matrix $\mathbf{S}$, the NMF requires also the mixing matrix $\mathbf{A}$. The mixing matrix can be approximated by a similar gradient descent algorithm:

$$a_{pn} \leftarrow a_{pn} + \eta_{pn}([\mathbf{X}\mathbf{S}^{\mathrm{T}}]_{pn} - [\mathbf{A}\mathbf{S}\mathbf{S}^{\mathrm{T}}]_{pn}) \tag{29}$$

To estimate both the source matrix $\mathbf{S}$ and the mixing matrix $\mathbf{A}$ a simple approach is to alternate the updating steps (equation 29) while enforcing the nonnegativity constraint (equation 13):

$$\begin{aligned} s_{nt} &\leftarrow [s_{nt} + \eta_{nt}([\mathbf{A}^{\mathrm{T}}\mathbf{X}]_{pn} - [\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{S}]_{nt})]_{+} \\ a_{pn} &\leftarrow [a_{pn} + \eta_{pn}([\mathbf{X}\mathbf{S}^{\mathrm{T}}]_{pn} - [\mathbf{A}\mathbf{S}\mathbf{S}^{\mathrm{T}}]_{pn})]_{+} \end{aligned} \tag{30}$$

where $[x]_{+} = \max(0, x)$. The alternating update is performed until convergence is achieved.

Gradient descent is a simple method for finding an optimal factorization, but has several drawbacks. The constant update step size makes it difficult for the algorithm to advance efficiently. For a small η, the convergence of the algorithm will be slow. For a too large η, the convergence is not necessarily stable. Thus, other algorithms have been built to achieve fast and reliable convergence for the factorization.

3.7 Multiplicative update algorithm

A popular way to circumvent the constraints of the gradient descent method is the multiplicative update method, presented by Lee and Seung in 2000 [34]. The multiplicative update algorithm redefines the update step variable η on each iteration, thus making the convergence faster than in static-η gradient descent. To minimize the NMF cost function in equation 14, η is defined after each step as

$$\eta_{nt} = \frac{s_{nt}}{[\mathbf{A}^{T}\mathbf{A}\mathbf{S}]_{nt}}. \tag{31}$$

Substituting η of gradient descent with η_{nt}, the update rules for A and S become

$$\begin{aligned} s_{nt} &\leftarrow s_{nt}\frac{[\mathbf{A}^{T}\mathbf{X}]_{pn}}{[\mathbf{A}^{T}\mathbf{A}\mathbf{S}]_{pn}} \\ a_{pn} &\leftarrow a_{pn}\frac{[\mathbf{X}\mathbf{S}^{T}]_{pn}}{[\mathbf{A}\mathbf{S}\mathbf{S}^{T}]_{pn}} \end{aligned} \tag{32}$$

To avoid division by zero, a small but positive value $\epsilon > 0$ can be added as shown in equation 33

$$
\begin{aligned}
s_{nt} &\leftarrow s_{nt} \frac{[\mathbf{A}^T\mathbf{X}]_{pn}}{[\mathbf{A}^T\mathbf{A}\mathbf{S}]_{pn} + \epsilon} \\
a_{pn} &\leftarrow a_{pn} \frac{[\mathbf{X}\mathbf{S}^T]_{pn}}{[\mathbf{A}\mathbf{S}\mathbf{S}^T]_{pn} + \epsilon}
\end{aligned}
\tag{33}
$$

3.8 Alternating Least Squares algorithm

Alternating Least Squares (ALS) algorithm is a quick and efficient method for computing NMF, and outperforms several methods in speed, especially for large-scale problems. The efficiency of the algorithm has been proven in several studies [1, 31]. The drawbacks of the standard ALS algorithm are proneness to converging to local, suboptimal solutions and susceptibility to noise. [1, 32, 31, 50].

The ALS algorithm minimizes the NMF cost function presented in equation 14, by setting one of the factorization matrices as a constant, and optimizing the other matrix by using the least square error method. The ALS algorithm is related to Expectation-Maximization or EM-algorithm, which is used to calculate maximum likelihood estimates. [7, 11].

The ALS algorithm solves the following minimization problems:

$$
\begin{aligned}
\mathbf{A}^{(t+1)} &= \arg\min_{\mathbf{A}} ||\mathbf{X} - \mathbf{A}\mathbf{S}^{(t)}||_F^2, \text{ s.t. } \mathbf{A} \geq 0, \\
\mathbf{S}^{(t+1)} &= \arg\min_{\mathbf{S}} ||\mathbf{X}^T - \mathbf{S}^T[\mathbf{A}^{(t+1)}]^T||_F^2, \text{ s.t. } \mathbf{S} \geq 0,
\end{aligned}
\tag{34}
$$

Unlike in the gradient descent method, the ALS technique fixes one of the factorization matrices and directly estimates the other. When one of the factorization matrices is fixed, the stationary points of the error in equation 14 are

$$
\begin{aligned}
\nabla_{\mathbf{A}} J_E(\mathbf{X}||\mathbf{A}\mathbf{S}) = \frac{\partial J_E(\mathbf{X}||\mathbf{A}\mathbf{S})}{\partial \mathbf{A}} &= \left[-\mathbf{X}\mathbf{S}^T + \mathbf{A}\mathbf{S}\mathbf{S}^T\right] = 0 \\
\nabla_{\mathbf{A}} J_E(\mathbf{X}||\mathbf{A}\mathbf{S}) = \frac{\partial J_E(\mathbf{X}||\mathbf{A}\mathbf{S})}{\partial \mathbf{A}} &= \left[-\mathbf{A}^T\mathbf{X} + \mathbf{A}^T\mathbf{A}\mathbf{S}\right] = 0
\end{aligned}
\tag{35}
$$

To avoid division by zero, negative entries of $\mathbf{A}$ and $\mathbf{S}$ are replaced by a small positive value, e.g. $\epsilon = 10^{-9}$.

When it is assumed that the estimated components are nonnegative, the update rules for the alternating least squares algorithm are

$$
\begin{aligned}
\mathbf{A} &\leftarrow \left[\mathbf{X}\mathbf{S}^T(\mathbf{S}\mathbf{S}^T)^{-1}\right]_+ \\
\mathbf{S} &\leftarrow \left[(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{X}\right]_+
\end{aligned}
\tag{36}
$$

3.9 Sparsity

Sparsity is a quality of data where most of the elements of data are zero, and the information is encoded by only a few elements. Text data presented through using a vector space model is a good example of sparse data, as each document contains only a very small subset of all terms in the available data.

Text data is by no means the only type of data that fulfills the nonnegativity requirement in addition to being sparse. In image processing and computer vision, nonnegative matrix factorization can be used in extracting relevant components from an image [27, 45]. The requirement of nonnegativity forces the component matrices to be interpretable, as the source signal is directly interpretable as a source image. Image processing can be expanded into color image processing by treating the color channels as a further color dimension. A further dimension into the time domain allows processing of video data. This also allows modelling of medical data, such as magnetic resonance imaging (MRI).

Another application sector for sparsity constrained nonnegative matrix factorization is economics. Several of the quantitative parameters of economic modelling, such as volume and price are non-negative [52]. In biology, gene expression research produces gene expression data that is both nonnegative and sparse. The research questions also often include source signal problems where sparsity controlled nonnegative matrix factorization is an useful approach [45]. Other application fields of sparsity controlled nonnegative matrix factorization are information retrieval and environmental science [7].

3.9.1 Measuring sparsity

There is no single, commonly accepted sparsity measure used in literature, but all measures should measure a vector as completely sparse if all of its energy is packed in a single value, and not sparse at all if all of its components are equally distributed. In this thesis, the sparsity measure proposed by Hoyer [26] will be used. This sparsity measure, $Sp(x)$, is based on the relationship between the L_1 and L_2 norms of a vector. The L_1 and L_2 norms are

$$\begin{aligned} L_1 &= \sum_{i=1}^{n} |x_i| \\ L_2 &= \sqrt{\sum_{i=1}^{n} x_i^2} \end{aligned} \tag{37}$$

and the sparsity measure is given as

$$Sp(\mathbf{x}) = \frac{\sqrt{n} - \left(\sum_{i=1}^{n} |x_i|\right) / \sqrt{\sum_{i=1}^{n} x_i^2}}{\sqrt{n} - 1}, \tag{38}$$

where n is the dimensionality of $\mathbf{x} = x_1, x_2, \ldots, x_n$. The measure can have values between zero and one, according to the criterion presented above. A vector $\mathbf{x}$ with an infinite amount of dimensions would have a $Sp(\mathbf{x}) = 1$ if only one element would contain a non-zero value. If the absolute value of each of the dimensions would be equal, $Sp(\mathbf{x})$ would be zero.

3.9.2 Controlling sparsity in NMF

Adding sparsity constraints to NMF raises the question of whether the sources, the mixing or both are sparse. The goal of using sparsity measures in NMF is to produce a factorization that realistically describes the features of the modeled data.

In the case of document text classification, a sparsity assumption has different meaning for the sources and mixing. If sources are assumed to be sparse, it would be implied that each "topic" can be constructed with a very small subset of all terms available. A non-sparse source vector would include a wide array of possible terms. As for the sparseness mixing matrix, it assumes that each document contains only a few topics. Similarly, a non-sparse mixing matrix would imply that the document includes several topics, their importance scaled with different weights, but present nevertheless.

The NMF sparsity constraints are formulated by Hoyer [26] as strict constraints, $S_{\mathbf{S}}$ and $S_{\mathbf{A}}$, for the sparsity of source matrix vectors $\mathbf{s}_i$ and mixing matrix vectors $\mathbf{a_i}$, respectively. Thus, the standard NMF problem presented in equation 13 becomes:

$$\begin{aligned} &\min_{\substack{\mathbf{A}\geq 0\\ \mathbf{S}\geq 0}} \mathcal{D}(\mathbf{X}; \mathbf{AS}) \\ &\mathcal{D}(\mathbf{X}; \mathbf{AS}) = \frac{1}{2}||\mathbf{X} - \mathbf{AS}||_F^2, \\ &\text{with constraints} \\ &Sp(\mathbf{a}_i) = S_a, \forall i \\ &Sp(\mathbf{s}_i) = S_s, \forall i, \end{aligned} \tag{39}$$

where $\mathbf{a}_i$ is the i:th column of $\mathbf{A}$. This approach requires the user to set three parameters: p, as the number of source components and S_a, S_s, as the required levels of sparsity for each factorization matrix. The algorithm for NMF, with these sparsity constraints uses the multiplicative update step presented by Lee and Seung[34]. The complete algorithm is thus

1. $\mathbf{A}$ and $\mathbf{S}$ are randomly initialized to positive matrices
2. If $Sp(\mathbf{a}_i) = S_a, \forall i$, project each $\mathbf{a}_i$ to be non-negative such that L_2 norm stays the same but L_1 norm is set to achieve the sparsity constraint
3. If $Sp(\mathbf{s}_i) = S_s, \forall i$, project each $\mathbf{s}_i$ to be non-negative such that L_2 norm stays the same but L_1 norm is set to achieve the sparsity constraint

4. Iterate

   (a) If $Sp(\mathbf{a}_i) = S_a, \forall i$

      i. $\mathbf{A} \leftarrow \mathbf{A} - \eta_{\mathbf{A}}(\mathbf{AS} - \mathbf{X})\mathbf{S}^{\mathrm{T}}$

      ii. Project each $\mathbf{a}_i$ to be non-negative such that L_2 norm stays the same but L_1 norm is set to achieve the sparsity constraint.

   else multiplicative step $\mathbf{A} \leftarrow \mathbf{A} \otimes \left(\mathbf{XS}^{\mathrm{T}}\right) \oslash \left(\mathbf{ASS}^{\mathrm{T}}\right)$

   (b) If $Sp(\mathbf{s}_i) = S_s, \forall i$

      i. $\mathbf{S} \leftarrow \mathbf{S} - \eta_{\mathbf{S}}\mathbf{A}^{\mathrm{T}}(\mathbf{AS} - \mathbf{X})$

      ii. Project each $\mathbf{s}_i$ to be non-negative such that L_2 norm stays the same but L_1 norm is set to achieve the sparsity constraint.

   else multiplivative step $\mathbf{S} \leftarrow \mathbf{S} \otimes \left(\mathbf{A}^{\mathrm{T}}\mathbf{X}\right) \oslash \left(\mathbf{A}^{\mathrm{T}}\mathbf{AS}\right)$

Operators $\otimes$ denotes elementwise multiplication of matrices, and $\oslash$ denotes elementwise division.

The projection of a vector $\mathbf{x}$ to a non-negative vector $\mathbf{s}$ such that $\mathbf{s} = \mathrm{argmin}_s||\mathbf{x} - \mathbf{s}||_E$. $\mathbf{s}$, given constant norms L_1 and L_2 is gained with the following algorithm presented by Hoyer [26].

1. Set $s_i := x_i + (L_1 - x_i)/\dim(\mathbf{x}), \forall i$
2. Set $Z := \{\ \}$
3. Iterate

   (a) Set $m_i := \begin{cases} L_1/(\dim(\mathbf{x}) - \mathrm{size}(Z)) \text{ if } i \notin Z \\ 0 \text{ if } i \in Z \end{cases}$

   (b) Set $\mathbf{s} := \mathbf{m} + \alpha(\mathbf{s} - \mathbf{m})$, where $\alpha \geq 0$ is selected such that the resulting $\mathbf{s}$ satisfies the L_2 norm constraint. This requires solving a quadratic equation.

   (c) If all components of $\mathbf{s}$ are non-negative, return $\mathbf{s}$, end

   (d) Set $Z := Z \cup \{i; s_i < 0\}$

   (e) Set $\mathbf{s}_i := 0, \forall Z$

   (f) Calculate $c := (s_i - L_1)/\dim(\mathbf{x}) - \mathrm{size}(Z))$

   (g) Set $s_i := s_i - c, \forall i \notin Z$

   (h) Go to (a)

The algorithm projects the given vector into a hyperplane with a set L_1 norm. Within the projection space, a new projection is performed to the closest point on the joint constraint hypersphere. The joint constraint hypersphere is the intersection of the L_1 sum and the L_2 norm constraints. If the result is completely non-negative, the algorithm has reached its destination and exits its iteration. If not, the components with negative values are set to zero and the iteration begins anew. [26]

3.10 Connection of NMF to other data factorization methods

Because of the strong theoretical framework behind NMF, systematic analysis about its connections to other factorization and clustering methods has been done. Here are presented the relations between NMF and k-means, spectral clustering and probabilistic latent semantic indexing.

3.10.1 Kernel K-means Clustering and NMF

Variations of NMF have been shown to be connected to k-means clustering. A variation of NMF, the symmetric NMF, factorizes a symmetric matrix $\mathbf{X} = \mathbf{X}^T$ into the product of a lower-dimensional matrix $\mathbf{A}$ and its transpose $\mathbf{A}^T$. [12]

K-means clustering is a widely used method that is based on minimizing the Euclidean distance error of the representation of the data with K cluster centroids. The error function to be minimized, J_K, can be formulated as

$$J_K = \sum_{k=1}^{K} \sum_{i \in C_k} ||\mathbf{x}_i - \mathbf{m}_k||^2 = c_2 - \sum_k \frac{1}{n_k} \sum_{i,j \in C_k} \mathbf{x}_i^T \mathbf{x}_j, \tag{40}$$

where $\mathbf{x}_i$ are the data vectors and $\mathbf{m}_k$ are the cluster centroids. $\sum_{i \in C_k} \mathbf{X}_i / n_K$ is the centroid of cluster C_k, containing n_k data points, and $c_2 = \sum_i ||\mathbf{x}_i||^2$. After clustering, the membership of each data point to cluster can be represented by K indicator vectors $\mathbf{h}_i, i = 1 \cdots K$. These indicator vectors contain zeroes except for data points belonging to that cluster, where the value is one. When a normalization constraint is added, the clustering matrix H can be formulated as

$$\begin{aligned} H &= (\mathbf{h}_1, \ldots, \mathbf{h}_K), \mathbf{h}_k^T \mathbf{h}_l = \delta_{kl} \\ \mathbf{h}_k &= \frac{(0, 1, 0, \ldots)^T}{(\sum_i \mathbf{h}_{ki})^{1/2}} \end{aligned} \tag{41}$$

Using matrix notation, this is written as

$$J_K = \mathrm{Tr}(\mathbf{X}^T \mathbf{X}) - \mathrm{Tr}(\mathbf{H}^T \mathbf{X}^T \mathbf{X} \mathbf{H}) \tag{42}$$

As the first term is constant, optimizing J_K can also be formalized as

$$\max_{\mathbf{H}^T \mathbf{H} = \mathbf{I}, \mathbf{H} \geq 0} J_W(\mathbf{H}) = \mathrm{Tr}(\mathbf{H}^T \mathbf{W} \mathbf{H}) \tag{43}$$

Kernel K-means is a variation of the K-means clustering method where the data X is mapped into a higher dimensional space by a mapping function $\mathbf{x}_i \to \phi(\mathbf{x}_i)$ before

clustering. The pairwise similarity matrix $\mathbf{X}^T\mathbf{X}$ is a standard inner-product kernel function. When it is applied on the k-means objective function J_K, the objective function is rewritten as

$$\min J_K(\phi) = \sum_i ||\phi(\mathbf{x}_i)||^2 - \sum_k \frac{1}{n_k} \sum_{i,j \in C_k} \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) \tag{44}$$

The first term will be a constant depending only on the kernel function ϕ and thus will not affect the optimization. Defining the kernel matrix $\mathbf{W}_{ij} = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_i)$ and using the cluster indicator matrix $\mathbf{H}$ the kernel K-means clustering can be written as

$$\begin{aligned} \min J_K = -\sum_k \frac{1}{n_k} \sum_{i,j \in C_k} \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) &= \sum_k \frac{1}{n_k} \sum_{i,j \in C_k} w_{ij} \\ &= \mathrm{Tr}(\mathbf{H}^T\mathbf{W}\mathbf{H}) \end{aligned} \tag{45}$$

The symmetric NMF objective function that minimizes the difference between a matrix $\mathbf{X}$ and its factorization $\mathbf{A}\mathbf{A}^T$ can be formulated as

$$\min J_1 = ||\mathbf{X} - \mathbf{A}\mathbf{A}^T||^2 \tag{46}$$

To prove the equality between kernel K-means clustering and the symmetric NMF, the equation 45 is written as

$$\begin{aligned} \mathbf{H} &= \underset{\mathbf{H}^T\mathbf{H}=\mathbf{I}, \mathbf{H} \geq 0}{\text{argmin}} - 2\mathrm{Tr}(\mathbf{H}^T\mathbf{W}\mathbf{H}) \\ &= \underset{\mathbf{H}^T\mathbf{H}=\mathbf{I}, \mathbf{H} \geq 0}{\text{argmin}} ||\mathbf{W}||^2 - 2\mathrm{Tr}(\mathbf{H}^T\mathbf{W}\mathbf{H}) + ||\mathbf{H}^T\mathbf{H}||^2 \\ &= \underset{\mathbf{H}^T\mathbf{H}=\mathbf{I}, \mathbf{H} \geq 0}{\text{argmin}} ||\mathbf{W} - \mathbf{H}\mathbf{H}^T||^2 \end{aligned} \tag{47}$$

The final form $\underset{\mathbf{H}^T\mathbf{H}=\mathbf{I}, \mathbf{H} \geq 0}{\text{argmin}} ||\mathbf{W} - \mathbf{H}\mathbf{H}^T||^2$ equals the symmetric NMF definition $\mathbf{X} = \mathbf{A}\mathbf{S}$ when the orthogonality requirement $\mathbf{H}^T\mathbf{H} = \mathbf{I}$ is dropped.

3.10.2 Probabilistic Latent Semantic Indexing

Probabilistic Latent Semantic Indexing (PLSI) [13] is a generative model for finding semantic components from text documents. The semantic components are latent statistical class models that can be interpreted as analoguous to source components of NMF. PLSI has been succesfully applied to document mining problems before

[25] and is shown in this chapter to optimize the same objective function as NMF with certain constraints.

PLSI maximizes an occurrence probability function J_{PLSI} that maximizes the likelihood of producing the data $\mathbf{X}$ that is used to train the method. In the case of PLSI, the data is rescaled so that the term-document matrix sums to one. Thus, for N documents containing M unique terms the word-document matrix is normalized with the total number of word occurrences: $\frac{[\mathbf{X}]_{ij}}{\sum_{ij}[\mathbf{X}]_{ij}}$.

For this proposition, the NMF model is formulated as $\mathbf{X} \approx \mathbf{AS}$, where the factorization matrices can be optimized by minimizing the cost function

$$J_{NMF} = \sum_{i=1}^{m}\sum_{j=1}^{n}[\mathbf{X}]_{ij}\log\frac{[\mathbf{X}]_{ij}}{[\mathbf{AS}]_{ij}} - [\mathbf{X}]_{ij} + [\mathbf{AS}]_{ij} \tag{48}$$

In comparison, PLSI maximizes a likelihood function

$$J_{PLSI} = \sum_{i=1}^{m}\sum_{j=1}^{n}[\mathbf{X}]_{ij}\log P(t_i, d_i) \tag{49}$$

where $P(t_i, d_j)$ is the factorized joint occurrence probability for the occurrence of term t_i in document d_j. An additional term z_k is defined such that it contains the dependencies between t_i and d_j. Thus, in

$$P(t_i, d_j) = \sum_{k} P(t_i|z_k)P(d_j|z_k)P(z_k) \tag{50}$$

the terms $P(t_i|z_k)$ and $P(d_j|z_k)$ are independent. The sums $\sum_{i=1}^{m} P(t_i|z_k)$, $\sum_{j=1}^{n} P(d_j|z_k)$ and $\sum_{k=1}^{K} P(z_k)$ are all normalized to 1.

To prove that L_1-normalized NMF and PLSI optimize the same objective function, PLSI objective function is written as

$$\min\sum_{i=1}^{m}\sum_{i=j}^{n} -[\mathbf{X}]_{ij}\log P(t_i, d_j) \tag{51}$$

A constant $\sum_{i=1}^{m}\sum_{i=j}^{n}[\mathbf{X}]_{ij}\log[\mathbf{X}]_{ij}$ is added:

$$\min\sum_{i=1}^{m}\sum_{i=j}^{n}[\mathbf{X}]_{ij}\log\frac{[\mathbf{X}]_{ij}}{P(t_i, d_j)} \tag{52}$$

Since

$$\sum_{i=1}^{m}\sum_{i=j}^{n}[P(t_i, d_j) - [\mathbf{X}]_{ij}] = [1 - 1] = 0 \tag{53}$$

the constant can be added to the PLSI objective function. Doing this, PLSI gets the format

$$\min \sum_{i=1}^{m} \sum_{i=1}^{n} [\mathbf{X}]_{ij} \log \frac{[\mathbf{X}]_{ij}}{P(t_i, d_j)} - [\mathbf{X}]_{ij} + P(t_i, d_j). \tag{54}$$

The nonnegative factorization $\mathbf{X} \approx \mathbf{AS}$ can be written as a product of normalized matrices $\hat{\mathbf{A}}$, $\hat{\mathbf{S}}$ and their diagonal weight matrix $\mathbf{Y}$ that has size of $k \times k$:

$$\begin{aligned} &\mathbf{AS} = \hat{\mathbf{A}}\mathbf{Y}\hat{\mathbf{S}}, \\ &\sum_{i=1}^{m} [\hat{\mathbf{A}}]_{ik} = 1, \\ &\sum_{j=1}^{m} [\hat{\mathbf{S}}]_{jk} = 1, \\ &\sum_{k=1}^{m} [\mathbf{Y}]_{kk} = 1 \end{aligned} \tag{55}$$

These matrices are equivalent to the normalization of probabilities in PLSI. Therefore $[\hat{\mathbf{A}}]_{ik} = P(t_i|z_k), [\hat{\mathbf{S}}]_{jk} = P(d_j|z_k), [\mathbf{Y}]_{kk} = P(z_k)$. By making this substitution in equation 54 becomes the objective function of NMF presented in equation 48. [13]

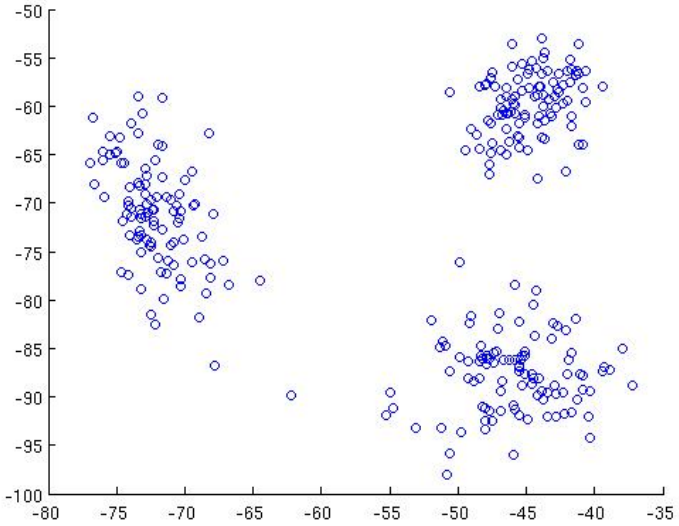

Figure 12: Two first principal components of multivariate gaussian data set generated by 6-dimensional gaussian distributions. PCA is used as a visualization for a higher than 2-dimensional data set.

4 Results

Nonnegative matrix factorization was used to cluster different kinds of data to evaluate its performance and suitability for neuroscience text mining. The first set of tests was run on a computer-generated multivariate gaussian toy data. The second set of tests was run on a standard corpus for text mining, the Reuters-21578 news article collection [4]. Finally, the text clustering was performed on a collection of neuroscience text excerpts from articles appeared in the journal Neuroimage.

4.1 Multivariate gaussian data

4.1.1 Data generation

The toy data used here to test the performance of classification methods was generated by Matlab's random number generator. The goal of data generation was to have multivariate data with clear clusters. This was achieved by setting a number of clusters, and defining a generative distribution for each of them. To gain a better view of the qualities of different classifiers, four different toy data sets were generated, with various levels of ratios between inter-cluster and intra-cluster variance.

The generative distributions were multivariate gaussians with random covariance matrices. The means of the distributions were sampled from a uniform distribution and all values below zero were set to zero, to maintain the nonnegativity required

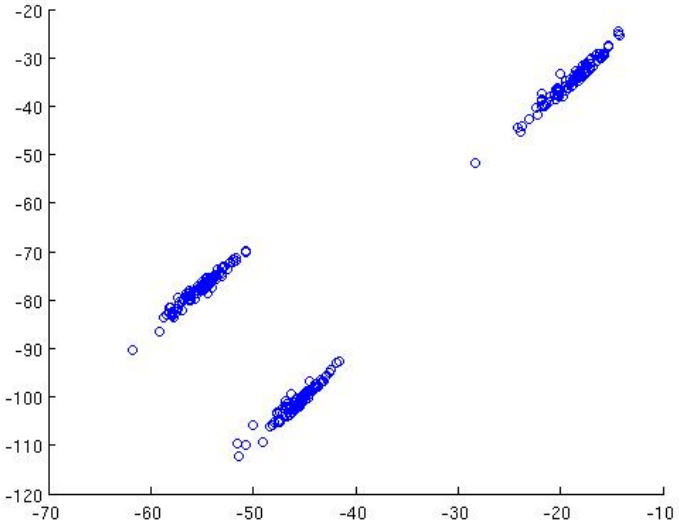

Figure 13: Two first principal components of the NMF reconstruction of the data shown in figure 12

by NMF. The first two principal components of a multivariate Gaussian data set generated in this way are presented in figure 12.

4.1.2 Clustering and Classification

The generated toy data visualized in figure 12 was classified using three different methods: k-means, nonnegative matrix factorization and k-means performed on the mixing matrix of a nonnegative matrix factorization of the data. The correct class for each data point was considered to be the index of the cluster that it was assigned to. To measure the performance of each classification method, a cross-validation was performed with one hundred repeats. For each repeat, the data sets were randomly divided into a training set consisting of 70% of the data points and a test set consisting of the rest of the data in that set.

The first of the classification methods was the classic k-means clustering. The cluster centroids were determined using the training data and the test data was classified by selecting the class of the closest centroid to each data point as the points' classification result.

The nonnegative matrix factorization was performed on the data sets, to reduce the dimensionality to five. The data was reconstructed using the NMF model $\hat{\mathbf{X}} = \mathbf{AS}$. The first two principal components of the data reconstruction of the data shown in figure 12 are visualized in figure 13. As visible in the visualization, NMF representation compresses the clusters found in the data, which makes classification significantly easier.

method	ϵ	ϵ_n	ϵ_p
k-means	0.087	0.031	0.017
NMF	0.28	0.040	0.057
NMF + k-means	0.57	0.023	.11

Table 1: The error measures by method. ϵ is the total ratio of false classifications. ϵ_n is the ratio of false negative classifications and ϵ_p the ratio of false positive classifications.

The simplest way to cluster data with NMF is to pick the NMF component with highest relevance factor and assign as the cluster or class of the data point. Thus, the maxima of rows of $\mathbf{A}$ will represent the clusters of data points. This approach interprets the columns of $\mathbf{S}$ as source signals for the data points.

$$\begin{aligned} \hat{\mathbf{x}}_i &= [\mathbf{AS}]_i \\ C_i &= \text{argmax}(\mathbf{a}_i) \end{aligned} \tag{56}$$

The third classification method used was a combination of NMF and k-means. First, the dimensionality of the data was reduced by NMF. This was done by modeling the data with standard NMF and using the mixing matrix $\mathbf{A}$ as the new feature matrix for the data points. As mentioned in section 4, NMF tends to tighten the clusters, which ideally improves the performance of k-means. In the performed tests, k-means without NMF performed still better than the alternatives.

4.2 Reuters-21578 data set

Reuters-21578 is a standard data set for text mining and categorization research. It contains 21578 news article items with topical labels and other meta-data , such as locations and people mentioned in the news article. The original set was compiled by the Carnegie Group, Inc. and Reuters, Ltd in 1987 and published in 1990. It has since become the most widely used data collection in text categorization research. [4]

The Reuters dataset documents are each several paragraphs long and inlcude news text from distinct fields such as politics and economy. Each document has one or more topic tags attached to it, which represent the document classes. The object of applying NMF and k-means clustering is to find structure from the data that corresponds to the real, human-defined topics.

A dendrogram of the Reuters dataset is presented in figure 14. As visible from the dendrogram, there are no easily distinguished clusterable structures in the dataset. This is at least partly due to the large number of dimensions in the data. As the Euclidean distance treats each dimension equally, in a sparse data set the distances are likely to be skewed.

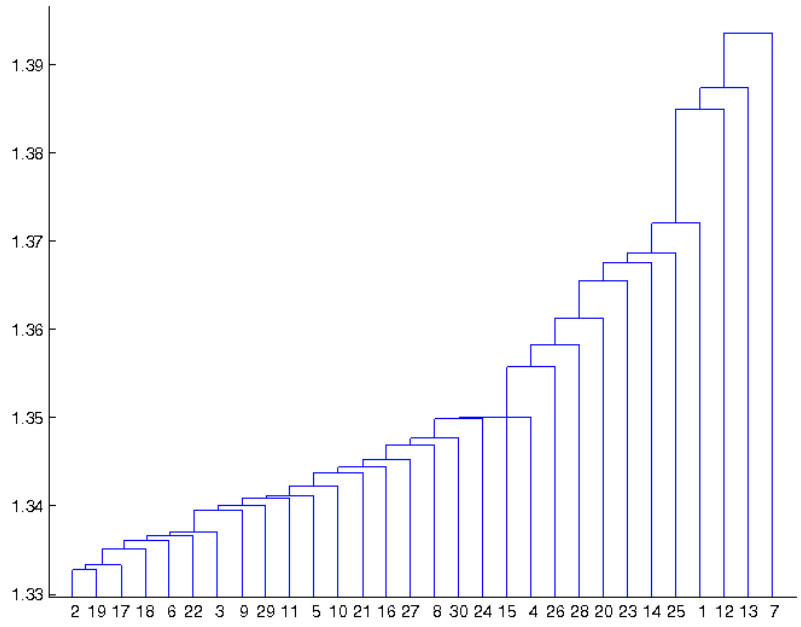

Figure 14: 135-level dendrogram of the Reuters dataset. The dendrogram was generated by comparing the Reuters news articles to each other by computing the Euclidean distance between the tf-idf feature rows. Each data point is declared a cluster. Then the clusters are joined by joining the pair of clusters with the shortest Euclidean distance between them. The new cluster will be located in the feature space as the average of the joined clusters. This dendrogram presents the joins from the point where all 135 clusters are separate to the "root" of the graph, where the last two clusters are joined.

Table 2: The Reuters dataset results by method. ϵ is the total ratio of false classifications. ϵ_n is the ratio of false negative classifications and ϵ_p the ratio of false positive classifications.

method	ϵ	ϵ_n	ϵ_p
k-means	0.46	0.092	0.092
NMF	0.58	0.12	0.15
NMF + k-means	0.46	0.092	0.092

The initial tests of the Reuters dataset are constructed picking only five topics, and making the dataset out of documents which contain given topics. This reduces the total number of documents to be processed and also gives five reasonably separable classes for the algorithm to work on. The five-topic dataset consists of articles with at least one of the following topic tags: acq for acquisitions, crude for crude oil, earn for earnings, trade for tradings and nat-gas for natural gas.

The classification of the 5-topic Reuters subset was done in three separate ways. First, classification is done with classical k-means. Then, the dimensionality of the original data was reduced using NMF and the processed data is clustered using k-means. Finally, the dimensionality of the original data was reduced using NMF to match the original number of classes in the data. This can be interpreted as representing each data point as a weighted combination of class components. The classification is done by selecting the component with highest weight as the point's class.

The results of the three different classification approaches are presented in table 2. The accuracy of a classification method was measured by using 70 percent of the items in the data set as training set and the rest as test set. The random division of the data was reperformed 10 times and the final classification measures are averages of those results.

4.3 Neuroimage text collection

The main goal of this study was to find out how the text mining methods presented here would manage to extract semantic information from neuroscience literature. A selection of scientific articles from the journal Neuroimage were used as test data. The methods presented in previous sections were tested by selecting the titles and abstracts from those articles as test data. The tf-idf feature extraction was performed on the titles and abstracts, following the same procedure as in the Reuters data set. Both the title and abstract data sets were clustered and the results compared to each other.

$E(n_s)$	$E(n_d)$	$\frac{E(n_s)}{(E(n_s)+E(n_d))}$
21.2	96.78	0.1798

Table 3: Neuroimage data set clustering results using k-means. The data was unlabeled, so the quality of the clustering method was done by comparing the similarity between clusterings performed using the titles and the abstracts. The quality of the method was estimated by comparing these two clusterings. From each clustering, the closest cluster centroids were declared to correspond each other. $E(n_s)$ is the mean of the distance between corresponding cluster centroids. $E(n_d)$ is the mean of the distance between non-corresponding cluster centroids. The quality measure $\frac{E(n_s)}{(E(n_s)+E(n_d))}$ is higher if the clustering of the title data and the abstract data resemble each other.

4.3.1 k-means

The data was investigated by using the k-means clustering algorithm. The results of this standardized clustering method will act as a baseline method. As a proof of concept, the dimensionality of the tf-idf data of the texts was dropped manually to only represent terms with known semantic meaning.

As the abstract- and title-data sets were clustered, these clusters needed to be labeled to evaluate the similarity of the two clusterings. The clusters were matched by computing an Euclidean distance matrix over the centroids of the clusters. Then the minimum of the matrix was searched and the respective clusters paired. The paired clusters were removed from the matrix and pairing continued until all the clusters had been paired for comparison.

By finding the corresponding clusters from the two clustering runs, the amount of data points labeled in the corresponding clusters was divided by the total amount of data points. The results of $n = 100$ clustering and labeling runs are presented in table 3.

4.3.2 NMF and k-means

This set of experiments was performed to study the effect of non-negative matrix factorization preprocessing on the performance of the k-means clustering. The measurement of performance and the data set were the same as in the Reuters tests.

Non-negative matrix factorization was used to extract components from the raw data and to reduce dimensionality. From the original 5-dimensional data with length n, the 3- dimensional source signal matrix A is extracted, and used as the feature vector. In an ideal case, the NMF is able to reduce the noise of the data and find 3 components that are able to express the information content of the data in a simpler and clearer form than without preprocessing.

There is a risk, however, that the reduction of dimensions will cause the clusters to overlap and thus reduce the accuracy of the clustering. Overlapping is a very real

$E(n_s)$	$E(n_d)$	$\frac{E(n_s)}{(E(n_s)+E(n_d))}$
15.84	102.16	0.13424

Table 4: Neuroimage data set clustering results using a combination of NMF and k-means. The method for measuring the clustering results is explained in the caption of table 3

risk as the final dimensionality of the data is lower than the number of clusters. In a situation where the clusters are evenly distributed in all five dimensions of the original data, collapsing the dimensionality may cause overlap and thus significantly weaken the results.

It is also possible that the NMF will not extract the source signals as expected, but instead keep the signal-to-noise ratio as it is or even worsen it.

As can be seen from table 4 that displays average results across the test runs, the clustering results were weaker when the data was preprocessed by using NMF dimensionality reduction. Reasons for this have been speculated above.

5 Discussion

5.1 Analysis of results

The clustering and classification on tf-idf text features data did not achieve results that would compare to a human-made classification. The question remains: Are the results good enough to be used in an application or to be further refined?

The reason why the tf-idf-based classification does not achieve a good accuracy might be the large information loss of the feature extraction. When only the term count is taken into account, the grammatical structure is lost, as well as the context. In addition, several terms are n-grams, where $n > 1$. In this study, only 1-grams or terms with a single word are counted as terms. Any meaningful terms with two consecutive words are thus lost.

Tf-idf feature extraction produces data with very high dimensionality. Tokenization and stemming can reduce the dimension of the data, but the cleaned data may still have tens of thousands of very sparsily populated dimensions. Furthermore, over 90% of a dimension might be zeroes, which may constitute a problem for many data mining tools. Hoyer et al. [26] have proposed an NMF processing method that would put a constraint on the sparsity of the factorization source matrices, but as this constraint is set as a parameter for the method, determining an efficient value for it would require considerable computational resources.

5.2 Future research

The purpose of this research was to contribute to a larger project, with the aim to utilize image as well as text data to automatically analyze the semantic meaning of neuroscience articles. The experiments performed in this thesis hint that satistically analyzing the semantic meaning produces promising results, as seen as well in previous research. By implementing those methods it is possible to reach classification and clustering results above random distribution of class labels.

Even though the tests explained in this thesis did not produce very good classification and clustering results, it is possible that improving the presented clustering methods and widening the selection of features improves the results. A 40% classification accuracy gained in clustering classic text mining data sets such as Reuters would not be high enough for a straightforward classification application, but it could be improved by the image analysis part of the application as well as the analysis of citation networks.

A suggestion for future research on automatic neuroscience article classifier would thus be the combination of results achieved by several different approaches. There's also clearly plenty of room for improvement in the text classification segment, as the published research has achieved significantly higher results with a different data set and different selection of features.

References

[1] Albright, R., Cox, J., Duling, D. Langville, A. N., Meyer, C. D.
Algorithms, initializations, and convergence for the nonnegative matrix factorization,
NCSU Technical Report Math 81706, 2006

[2] Altman, R. et al.
Text mining for biology - the way forward: opinions from leading scientists,
Genome Biology, vol 9(Suppl 2):S7, 2008

[3] Antezana, E., Kuiper, M., Mironov, V
Biological knowledge management: the emerging role of the Semantic Web technologies,
Briefings in Bioinformatics, vol 10, Issue 4, pp. 392-407, 2009

[4] Asuncion, A., Newman, D. J.
Reuters-21578 dataset,
UCI Machine Learning Repository Irvine, CA: University of California, School of Information and Computer Science, 2007
http://www.ics.uci.edu/∼mlearn/MLRepository.html

[5] Biber, D., Conrad, S., Reppen, R.
Corpus linguistics, Investigating language Structure and Use,
Cambridge, Cambridge UP 2008

[6] Bishop C. M.
Pattern Recognition and Machine Learning, Springer 2006

[7] Cichocki, A., Zdunek, R., Phan, A. H., Amari, S.
Nonnegative Matrix and Tensor Factorizations,
John Wiley & Sons, 2009

[8] Chomsky, N.
Syntactic Structures, Mouton & Co. 1957

[9] Cohen, A. M.
A survey of current work in biomedical text mining,
Briefings in Bioinformatics vol 6, Issue 1, pp. 57-71, 2005

[10] Crasto C. J., Marenco L. N., Migliore M., Mao B., Nadkarni P. M., Miller P., Shepherd G. M.
Text Mining Neuroscience Journal Articles to Populate Neuroscience Databases,
Neuroinformatics, vol 1, Issue 3 pp. 215-37, 2003

[11] Dempster, A. P., Laird, N. M., Rubin, D. B.
Maximum Likelihood from Incomplete Data via the EM algorithm,

Journal of the Royal Statistical Society, Series B (Methodological), Vol. 39, Issue 1., pp. 1-38, 1977

[12] Ding, C., He, X., Simon, H.
On the equivalence of Nonnegative Matrix Factorization and Spectral Clustering,
Proc. SIAM Data Mining Conference, 2005

[13] Ding, C., Li, T., Peng, W.
On the Equivalence between Non-negative Matrix Factorization and Probabilistic Latent Semantic Indexing,
Computational Statistics and Data Analysis vol 52, pp. 3913-3927, 2008

[14] Ding, C., Li, T., Peng, W., Park, H.
Orthogonal nonnnegative tri-factorizations for clustering,
KDD06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge Discovery and Data Mining, pp. 126-135, New York, NY, USA, 2006

[15] Feinerer, I., Hornik, K., Meyer, D.
Text Mining Infrastructure in R,
Journal of Statistical Software vol 25, Issue 5, March 2008
Package available at http://www.jstatsoft.org/v25/i05/

[16] Fisher, R. A.
The Use of Multiple Measurements in Taxonomic Problems.
Annals of Eugenics vol 7, Issue 2 pp. 179188, 1936

[17] Gao, Y., Church, G.
Improving molecular cancer class discovery through sparse non-negative matrix factorization,
Bioinformatics vol 21, Issue 21, pp. 3970-3975, 2005

[18] Georgescu, B., Shimshoni, I., Meer, P.
Mean shift based clustering in high dimensions: a texture classification example
Proc. 9th IEEE International Conference on Computer Vision (ICCV 2003) vol 1 pp. 456-463, Nice, France, 2003

[19] Gerstein, M., Seringhaus, M., Fields, S.
Structured digital abstract makes text mining easy,
Nature, Issue 447 pp. 142, 2007

[20] Gobinet, C., Elhafid, A., Vrabie, V., Huez,R., Nuzillard, D.
About importance of positivity constraint for source separation in fluorecence spectroscopy,
Proc. European Signal Processing Conference (EUSICPO 2005) Antalya, Turkey, 2005

[21] Gobinet, C., Perrin, E., Huez, R.
Application of nonnegative matrix factorization to fluorecence spectroscopy,
Proc. European Signal Processing Conference (EUSICPO 2004) Vienna, Austria, 2004

[22] Golub, G. H., Reinsch, C.
Singular value decomposition and least squares solutions,
Numerische Mathematik, vol 14, Issue 5, pp. 403-420, 1970

[23] Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Drowning, J. R., Caliguiri, M. A.
Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring,
Science Issue 286, pp. 531-537, 1999

[24] Hahn, U., Wermter, J., Blasczyk, R., Horn, P. A.
Text mining: powering the database revolution,
Nature, Issue 448, pp. 130, 2007

[25] Hofmann, T.
Probabilistic Latent Semantic Analysis,
Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence pp. 289-296, 1999

[26] Hoyer, P. O.
Non-negative Matrix Factorization with Sparseness Constraints,
Journal of Machine Learning Research vol 5, pp. 1457-1469, 2004

[27] Hunt, S. D., Rosario-Torres S., Vlez-Reyes M., Jimnez L.O.
New developments and application of the UPRM MATLAB hyperspectral image analysis toolbox. *Proceedings of SPIE: Algorithms and Technologies for Multi-spectral, Hyperspectral and Ultraspectral Imagery* vol 6565 of XXII, May 2007

[28] Ihaka, R., Gentleman, R.
R: A Language for Data Analyses and Graphics,
Journal of Computational and Graphical Statistics, vol 5, Issue 3, pp. 299-314, 1996

[29] Jolliffe, I. T.
Principal Component Analysis

[30] Kim J.-D., Ohta T., Tateisi Y., Tsujii J.
GENIA corpus - a semantically annotated corpus for bio-textmining,
Bioinformatics vol 19, Issue suppl 1 pp. 180-182, 2003

[31] Langville A. N., Meyer C. D., Albrtight R.
Initializations for the nonnegative matrix factorization,

Proceedings of the Twelth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, USA, August 20-23 2006

[32] Hancewicz, T. M., Wang, J.-H.
Discriminant image resolution: a novel multivariate image analysis method utilizing a spatial classification constraint in addition to bilinear nonnegativity,
Chemometrics and Intelligent Laboratory Systems vol 77, pp. 18-31, 2005

[33] Lee, D. D., Seung, H. S.
Learning the parts of objects by non-negative matrix factorization,
Nature, vol 401(6755), pp. 788-791, 1999

[34] Lee, D. D., Seung, H. S.
Algorithms for Non-negative Matrix Factorization,
Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference, The MIT Press, pp. 556-562, 2001

[35] Madsen, M. W.
The Limits of Machine Translation,
Thesis submitted for the degree of Master in Information Technology and Cognition, Department of Scandinavian Studies and Linguistics, Faculty of Humanities, University of Copenhagen

[36] Manning, C. D., Raghavan, P., Schütze H.
Introduction to Information Retrieval, Cambridge University Press, 2008

[37] Manning, C. D., Schütze, H.
Foundations of Statistical Natural Language Processing, The MIT Press, 2003

[38] Müller, H. M., Rangarajan A., Teal T. K., Sternberg P. W.
Textpresso for neuroscience: searching the full text of thousands of neuroscience research papers,
Neuroinformatics September volume 6(3): pp. 195-294, 2008

[39] Editorial: The database revolution
Nature, nr 445, pp. 229-230, 2007

[40] Ng, A. Y., Jordan, M. I., Weiss, Y.
On Spectral Clustering: Analysis and an Algorithm,
Neural Information Processing Systems 14, 2002

[41] Paatero, P., Tapper, U.
Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values,
Environmetrics 5, pp. 111-126, 1994

[42] Pauca V. P., Piper J., Plemmons R. J.
Nonnegative matrix factorization for spectral data ,
Linear Algebra and its Applications vol 416, Issue 1, pp. 29-47, 2006

[43] Porter, M. F.
An algorithm for suffix stripping,
Program vol 3, Issue 14 pp. 130-137

[44] Turing, A.
Computing Machinery and Intelligence,
Mind LIX (236), pp. 433-460. 1950

[45] Sajda P., Du S., Brown T.R., Parra L.C., Stoyanova R.
Recovery of constituent spectra in 3D chemical shift imaging using nonnegative matrix factorization.
Proc. of 4th International Symposium on Independent Component Analysis and Blind Signal Separation pp. 71-76, Nara, Japan, April 2003

[46] Sakakibara, Y., Brown, M., Hughey, R., Mian, I. S. et al.
Stochastic context-free grammars for tRNA modelling,
Nucleic Acids Research, Issue 22 pp. 5112-5120, 1994

[47] Saper, E.
Language: An Introduction to the Study of Speech,
New York: Harcourt Brace, 2004

[48] Shahnaz, F., Berry, M. W., Pauca V. P., Plemmons R. J.
Document clustering using nonnegative matrix factorization,
Information Processing & Management vol 41, Issue 2 pp. 373-386, 2006

[49] Sinclair, J.
The automatic analysis of corpora,
Directions in Corpus LInguistics (Proceedings of Nobel Symposium 82), 1992

[50] Wang, J. H., Hopke, P. K., Hancewicz, T. M., Zhang S.-L.
Application of modified alternating least squares regression to spectroscopic image analysis,
Analytica Chimica Acta 476, pp. 93-109, 2003

[51] Willett, P.
The Porter stemming algorithm: then and now,
Program: electroni library and information systems vol 40, Issue 3, pp. 219-223

[52] Yoo J., Choi S.
Orthogonal nonnegative matrix factorization: Multiplicative updates on Stiefel manifolds,
Intelligent Data Engineering and Automated Learning IDEAL 2008 vol 5326 of *Lecture notes in Computer Science* pp. 140-147, Springer-Verlag, Berlin 2008

[53] Zweigenbaum, P., Denmer-Fushman, D., Yu, H., Cohen, K. B.
Frontiers of biomedical text mining: current progress,
Briefings in Bioinformatics vol 8, Issue 5, pp. 358-375, 2007