Document Clustering Using Nonnegative Matrix Factorization

Farial Shahnaz and Michael W. Berry

Department of Computer Science, University of Tennessee, Knoxville, TN 37996-3450

V. Paul Pauca and Robert J. Plemmons ⋆

Department of Computer Science, Wake Forest University, Winston-Salem, NC 27109

Abstract

A methodology for automatically identifying and clustering semantic features or topics in a heterogeneous text collection is presented. Textual data is encoded using a low rank nonnegative matrix factorization algorithm to retain natural data nonnegativity, thereby eliminating the need to use subtractive basis vector and encoding calculations present in other techniques such as principal component analysis for semantic feature abstraction. Existing techniques for nonnegative matrix factorization are reviewed and a new hybrid technique for nonnegative matrix factorization is proposed. Performance evaluations of the proposed method are conducted on a few benchmark text collections used in standard topic detection studies.

Key words: conjugate gradient, constrained least squares, nonnegative matrix factorization, text mining.

1 Introduction

Text mining refers to the detection of trends, patterns, or similarities in natural language text. Given a collection of text documents, often the need arises

⋆ Research of co-authors Pauca and Plemmons was supported in part by the Air Force Office of Scientific Research under grant FA49620-03-1-0215, and by the Army Research Office under grant DAAD19-00-1-0540.

Email addresses: [shahnaz,berry]@cs.utk.edu (Farial Shahnaz and Michael W. Berry), [paucavp,plemmons]@wfu.edu (V. Paul Pauca and Robert J. Plemmons).

to classify the documents into groups or clusters based on similarity of content. For a relatively small collection, it may be possible to manually perform the partitioning of documents into specific categories. But to partition large volumes of text, the process would be extremely time consuming. Moreover, automation also greatly reduces the time needed to perform the classification.

When the categories or topics for classification are predefined, the process of classification is considered *supervised*; there are several methods in use that satisfactorily automate the task of supervised classification [3]. However, in absence of any information regarding the nature of the data, the problem of classification becomes much more difficult. For *unsupervised* classification of text data, only one valid assumption can be made, which is that the text collection is completely unstructured. The task then becomes organizing the documents into a structure based solely on patterns learned from the collection itself. This structure can be *partitional* or *hierarchical* [3]. The hierarchical organization of documents has a tree-like structure with the entire collection situated at the root level. In subsequent levels of the tree, the collection is partitioned into smaller groups and eventually each document is represented as a separate group at the bottom level.

If the text collection is given a partitional structure, then the documents in the collection are flatly partitioned or clustered into groups that are non-overlapping. The proposed Nonnegative Matrix Factorization (NMF) method for text mining introduces a technique for partitional clustering that identifies semantic features in a document collection and groups the documents into clusters on the basis of shared semantic features. The factorization can be used to compute a low rank approximation of a large sparse matrix along with preservation of natural data nonnegativity.

In the *vector space model* of text data, documents are encoded as n-dimensional vectors where n is the number of terms in the dictionary, and each vector component reflects the importance of the corresponding term with respect to the semantics of a document [1]. A collection of documents can, thus, be represented as a term-by-document matrix. Since each vector component is given a positive value (or weight) if the corresponding term is present in the document and a null or zero value otherwise, the resulting term-by-document matrix is always nonnegative. This inherent data nonnegativity is preserved by the NMF method as a result of constraints (placed on the factorization) that produce nonnegative lower rank factors that can be interpreted as semantic features or patterns in the text collection. The vectors or documents in the original matrix can be reconstructed by combining these semantic features, and documents that have common features can be viewed as a cluster. As shown by Xu et al. [15], NMF outperforms traditional vector space approaches to information retrieval (such as latent semantic indexing) for document clustering on a few topic detection benchmark collections.

2 Related Work

Nonnegative matrix factorization differs from other rank reduction methods for vector space models in text mining, e.g., principal component analysis (PCA) or vector quantization (VQ), due to use of constraints that produce nonnegative basis vectors, which make possible the concept of a *parts-based representation* [8]. Lee and Seung first introduced the notion of parts-based representations for problems in image analysis or text mining that occupy nonnegative subspaces in a vector-space model. Techniques like PCA and VQ also generate basis vectors – various additive and subtractive combinations of which can be used to reconstruct the original space. But the basis vectors for PCA and VQ contain negative entries and cannot be directly related to the the original vector space to derive meaningful interpretations. In the case of NMF, the basis vectors contain no negative entries – this allows only additive combinations of the vectors to reproduce the original. So the perception of the whole, be it an image or a document in a collection, becomes a combination of its parts represented by these basis vectors. In text mining, the vectors represent or identify semantic features, i.e., a set of words denoting a particular concept or topic. If a document is viewed as a combination of basis vectors, then it can be categorized as belonging to the topic represented by its principal vector. Thus, NMF can be used to organize text collections into partitional structures or clusters directly derived from the nonnegative factors.

Recently Xu et al. [15] have demonstrated that NMF outperforms methods such as singular value decomposition and is comparable to graph partitioning methods that are widely used in clustering text documents. The tests were conducted on two different datasets: the Reuters data corpus [1] and TDT2 corpus [2], both considered benchmark collections for topic detection. These two data corpora are also used in this study to observe the results of using nonnegative factorization for text mining or document clustering. The algorithm used to derive the factorization introduces a new parameter to control the number of basis vectors used to reconstruct the document vectors, thereby providing a mechanism to balance the tradeoff between accuracy and computational cost (including storage).

[1] Reuters-21578 at `http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html`.

[2] `http://www.lcd.upenn.edu`.

3 Algorithm

With the standard vector space model, a set of documents S can be expressed as a $m \times n$ matrix V, where m is the number of terms in the dictionary and n is the number of documents in S. Each column V_j of V is an encoding of a document in S and each entry v_{ij} of vector V_j is the significance of term i with respect to the semantics of V_j, where i ranges across the terms in the dictionary. The NMF problem is defined as finding a low rank approximation of V in terms of some metric (e.g., the norm) by factoring V into the product (WH) of two reduced-dimensional matrices W and H. Each column of W is a basis vector, i.e., it contains an encoding of a semantic space or concept from V and each column of H contains an encoding of the linear combination of the basis vectors that approximates the corresponding column of V. Dimensions of W and H are $m \times k$ and $k \times n$ respectively, where k is the reduced rank or selected number of topics. Usually k is chosen to be much smaller than n, but more accurately, $k << min(m, n)$. Finding the appropriate value of k depends on the application and is also influenced by the nature of the collection itself [5].

Common approaches to NMF obtain an approximation of V by computing a (W,H) pair to minimize the Frobenius norm of the difference $V - WH$. The problem can be cast in the following way [11] – let $V \in R^{m \times n}$ be a nonnegative matrix and $W \in R^{m \times k}$ and $H \in R^{k \times n}$ for $0 < k << min(m, n)$. Then, the objective function or minimization problem can be stated as

$$\min_{W,H} \|V - WH\|_F^2, \tag{1}$$

with $W_{ij} > 0$ and $H_{ij} > 0$ for each i and j.

The matrices W and H are not unique. Usually H is initialized to zero and W to a randomly generated matrix where each $W_{ij} > 0$ and these initial estimates are improved or updated with alternating iterations of the algorithm. In the following subsections some existing NMF techniques are discussed and a new algorithm is proposed.

3.1 Multiplicative method

The NMF method proposed by Lee and Seung is based on multiplicative update rules of W and H. This scheme is referred to as the multiplicative method (MM). Algorithm 3.1.1 contains a formal statement of the method [11].

MM Algorithm

(1) Initialize W and H with nonnegative values.
(2) Iterate for each c, j, and i until convergence or after l iterations:
  (a) $H_{cj} \leftarrow H_{cj} \frac{(W^T V)_{cj}}{(W^T W H)_{cj} + \epsilon}$
  (b) $W_{ic} \leftarrow W_{ic} \frac{(V H^T)_{ic}}{(W H H^T)_{ic} + \epsilon}$

In steps 2(a) and (b), ϵ, a small positive parameter equal to 10^{-9}, is added to avoid division by zero. As observed from the MM Algorithm, W and H remain nonnegative during the updates. Simultaneous updating of W and H generally yield better results than updating each matrix factor fully before the other. In the algorithm, the columns of W or the basis vectors are normalized at each iteration; in case of W, the optimization is performed on a unit hypersphere with the columns of W effectively being mapped to the surface of the hypersphere by repeated normalization [11].

The computational complexity of MM can be shown to be $O(kmn)$ operations (for a rank-k approximation) per iteration [11]. Once the term-by-document matrix V has been factored into W and H, if new data needs to be added, then the data can be a direct addition to W with a minor modification to H if k is not fixed. In case of a fixed k, the new data can be integrated by further iterations with W and H as the initial approximations. In [8] it is shown by Lee and Seung that under the MM-update rules, the objective function (1) is monotonically non-increasing and becomes constant if and only if W and H are at a stationary point. This multiplicative method is related to expectation-maximization approaches used in image restoration, e.g. [12], and can be classified as a diagonally-scaled gradient descent method [5].

3.2 Sparse Encoding

A new nonnegative sparse encoding scheme, based on the study of neural networks has been suggested by Hoyer [6]. This scheme is applicable to the decomposition of datasets into independent feature subspaces by Hyvärinen and Hoyer [7]. The method proposed by Hoyer [6] has an important feature that enforces a statistical sparsity of the H matrix. As the sparsity of H increases, the basis vectors become more localized, i.e., the parts-based representation of the data in W become more and more enhanced. Mu, Plemmons and Santago [10] have put forth a regularization approach that achieves the same objective of enforcing statistical sparsity of H by using a point-count regularization scheme that penalizes the number of non-zero entries rather than the sum of entries $\sum_{ij} H_{ij}$ in H.

3.3 A Hybrid Method

The NMF algorithm used in this study [11] is a hybrid method that combines some of the better features of the methods discussed in the previous sections. In this approach, the multiplicative method, which is basically a version of the gradient descent optimization scheme, is used at each iterative step to approximate the basis vector matrix W. H is calculated using a constrained least squares (CLS) model as the metric. It serves to penalize the non-smoothness and non-sparsity of H; as a result of this penalization, the basis vectors or topics in W become more localized, thereby reducing the number of vectors needed to represent each document. The method for approximating H is similar to the methods described in [6] and [10] and related to the least squares Tikhonov regularization technique commonly used in image restoration [12]. This hybrid algorithm is denoted by GD-CLS (gradient descent with constrained least squares) in [11].

GD-CLS Algorithm

(1) Initialize W and H with nonnegative values, and scale the columns of W to unit norm.
(2) Iterate until convergence or after l iterations:
   (a) $W_{ic} \leftarrow W_{ic} \frac{(VH^T)_{ic}}{(WHH^T)_{ic} + \epsilon}$, for c and i $[\epsilon = 10^{-9}]$
   (b) Rescale the columns of W to unit norm
   (c) Solve the constrained least squares problem:

$$\min_{H_j} \{\|V_j - WH_j\|_2^2 + \lambda \|H_j\|_2^2\},$$

where the subscript j denotes the j^{th} column, for $j = 1, \ldots, m$. Any negative values in H_j are set to zero. The parameter λ is a regularization value that is used to balance the reduction of the metric

$$\|V_j - WH_j\|_2^2$$

with enforcement of smoothness and sparsity in H.

4 Software Implementation

Two software packages, namely GTP and LAPACK, were used in the C-based implementation of GD-CLS algorithm. The General Text Parser (GTP) is a software environment developed at the University of Tennessee by Giles et al.

[4]. One of the functions of GTP is to parse text documents and construct a sparse matrix data structure, i.e., a term-by-document matrix that defines the relationship between the documents and the parsed terms [9]. The GTP software can be used to parse single files or entire directories and is fitted with the capability to process both raw text and HTML files. The user can also integrate external filters into the software to process other forms of tagged data. Currently there are two versions of the software available – one in C++ and another in Java – both of which are designed to facilitate users with all ranges of expertise. For this study, the C++ version of GTP was used.

LAPACK has various routines, which can be individually downloaded from the LAPACK website [3], for solving different types of linear equations. For the C version of NMF, the *dposv* software routine of LAPACK is used to derive solutions (in double precision) to linear systems of the form $AX = B$, where A is a symmetric positive definite matrix.

5 Experiments

Originally written in MATLAB (see [11] and Figure 1) the proposed NMF algorithm or GD-CLS has been converted to C in this study for scalability. Performance evaluations are conducted using two different datasets — the Reuters Document Corpus and TDT2. This section describes the methodology used for evaluation, while the actual results [4] are discussed in Section 6.

5.1 Reuters

The Reuters data corpus [5], contains 21578 documents and 135 topics or document clusters created manually and each document in the corpus is been assigned one or more topics or category labels based on its content. The manually created cluster sizes, i.e., the number of documents assigned to the topics, range anywhere from less than ten to nearly four thousand topics. The documents are in SGML format (see [13]) with meta tags denoting title, topic(s), and beginning and end of content.

For this experiment, documents associated with only one topic are used and topics with cluster sizes smaller than five are discarded. To achieve this, a Perl

[3] `http://www.netlib.org/lapack`

[4] All results are collected on a Sun Microsystems SunBlade 1000 workstation with 500 MHz UltraSPARC-IIe processor, 256KB L2 cache, 512MB DRAM and 20GB internal disk.

[5] `http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html.`

```
[W, H] = gdcls(V, k, maxiter, lambda, options)
myeps = 10^-9;
if strcmp(options, 'nonneg')
   neg = 1;
else
   neg = 0;
end
[m,n] = size(V);
W = rand(m, k);
H = zeros(k, n);
for j = 1 : maxiter,
    A = W' * W + lambda * eye(k);
    for i = 1 : n
      b = W' * V(:,i);
     H(:,i) = A / b;
    end
   if neg == 1
     H = H .* (H > 0);
    end
   W = W .* (V * H') ./ (W * (H * H') + myeps);
end
```

Fig. 1. MATLAB implementation of GS-CLS Algorithm

script (see [13]) is used to traverse through the corpus and create an index of topics with associated cluster sizes, where a document is considered part of a cluster only if it has a single topic.

In order to observe the performance of the GD-CLS implementation of NMF as the complexity of the problem increases, i.e., as the number of clusters or the parameter k is incremented, seven different k values 2, 4, 6, 8, 10, 15, 20 are chosen. For each k, three different document collections or subsets are generated by the filter using different topic files, which result in creation of three term-by-document (sparse) matrices for each k. As the predominant number of elements of these matrices are zero, the Harwell-Boeing (HB) column-compressed sparse matrix format [2] is used to access all non-zero elements. After the HB matrices are generated, the NMF clustering algorithm is performed on all 21 matrices ($7k$ values $\times$ 3 document subsets each) to produce the W and H factors for each HB matrix. For any given HB matrix V, with k topics and n documents, matrix W has k columns or basis vectors that represent the k clusters, while matrix H has n columns that represent the n documents. A column vector in H has k components, each of which denotes the contribution of the corresponding basis vector to that column or document. The classification or clustering of documents is then performed based on the index of the highest value of k for each document. So, for document i

$(i = 1, \ldots, n)$, if the maximum value is the jth entry $(j = 1, \ldots, k)$, document i is assigned to cluster j. After the documents are clustered into k topics, the NMF generated k clusters are compared to the original k clusters using a mapping function. The mapping is performed using a Perl script that assigns the original cluster labels to the NMF clusters based on a similarity measure. The example below provides an explanation of the mapping process for $k = 2$.

Once the relabeling is accomplished, the accuracy of the classification or clustering is assessed using the metric AC [15] defined by

$$\text{AC} = \sum_{i=1}^{n} \delta(d_i)/n,$$

where $\delta(d_i)$ is set to 1 if d_i has the same topic label for both NMF and the original classification, and set to 0 otherwise, and n is the number of documents in the collection. So, for our two-cluster example

$$\begin{aligned} \text{AC} &= \{\delta(d_1) + \delta(d_2) + \delta(d_3) + \delta(d_4) + \delta(d_5)\}/5 \\ &= \{1 + 1 + 1 + 1 + 0\}/5 = 4 \ / \ 5 = 0.8. \end{aligned}$$

In the GD-CLS implementation of NMF, the contribution of the λ parameter with which the sparsity of H is controlled is also of interest. Hence, for each k, results for three different λ values (0.1, 0.01, 0.001) are calculated.

Two-Cluster Example.

Original Topic Set $T = \{A, B\}$
Document subset $D = \{d_1, d_2, d_3, d_4, d_5\}$
$Cluster_A = \{d_2, d_3\}$, $Cluster_B = \{d_1, d_4, d_5\}$

Using GD-CLS on the HB matrix generated from D with topic set T yields WH, where $W \in R^{m \times 2}$ and $H \in R^{2 \times 5}$. Assuming H has the value shown in Table 1, the clustering based on the maximum column entry is

$Cluster_1 = \{d_2, d_3, d_5\}$,
$Cluster_2 = \{d_1, d_4\}$.

The values of the following mapping function are used to form a matrix S (Table 2), where $S_{iX} = similarity(Cluster_i, Cluster_X)$ = number of documents in $Cluster_i$ that appear in $Cluster_X$, $i = (1, 2)$ and $X = \{A, B\}$.

Each $Cluster_i$ is assigned the original cluster label to which it is the most similar. $Cluster_1$ and $Cluster_2$ are assigned labels A and B respectively and the documents are reassigned to topics based on the new clustering. A comparison of the original clustering to the GD-CLS generated cluster labels is shown in Table 3.

Table 1
The 2×5 H matrix for the two-cluster example
(maximum entries are represented in boldface).

d_1	d_2	d_3	d_4	d_5
0.3	**1.2**	**0.2**	0.01	**2.1**
1.4	0.9	0.01	**1.4**	1.9

Table 2
The S matrix for the two-cluster example.

	$Cluster_A$	$Cluster_B$
$Cluster_1$	2	1
$Cluster_2$	0	2

Table 3
Comparison between original cluster labels and GD-CLS generated labels for the two-cluster example.

Document	Original label	GD-CLS label
d_1	B	B
d_2	A	A
d_3	A	A
d_4	B	B
d_5	B	A

5.2 TDT2

The second data corpus TDT2, obtained from the Language Data Consortium at The University of Pennsylvania [6], contains transcripts from a total of six news sources [7] in 3440 files, with each file containing several transcripts or documents. Although the corpus consists of about sixty-four thousand documents in SGML format (see [13]), some fourteen thousand of these are actually assigned a topic label and the rest are not classified. Among the preclassified documents, 7919 documents are single topic documents, i.e., these documents only have a single topic or category label. The SGML markup tags for each document denote a unique document ID or identification number and the beginning and end of text content. The document-topic relationships are described in a separate file that contains a line in it for each document with a category label. A line corresponding to a particular document consists of the document ID, topic label, and the name of the file containing that document.

In order to make the document collection from this corpus comparable to the Reuters dataset, some preprocessing with the use of Perl scripts is applied to the SGML files. First, the file containing the document-topic relationships is parsed and a *topic file* or a file containing a list of 73 topics that have cluster sizes of at least five documents is created. Here also, as with the Reuters collection, documents containing multiple topic labels are not deemed relevant. Since the entire document corpus consists of 64,000 documents and only 7919 are relevant to the experiments, another preprocessing step is taken to reduce the runtime of GTP by traversing the entire collection once and writing the relevant documents to a single file. For all subsequent testing, only this file is then used in order to avoid traversing thousands of irrelevant documents for each test run. Once the topic file and the reduced set of 7919 documents are at hand, several subsets are created to monitor the decline of accuracy for the NMF algorithm as complexity or the k values increase. As before, 7 different k values (2, 4, 6, 8, 10, 15, 20) are chosen with 10 different topic sets or document subsets each. After application of GD-CLS and the accuracy metric, this selection of datasets produces results presented in the following section.

6 Observations

The results from TDT2 and Reuters data corpora bring to attention trends such as the decline in accuracy in relation to the increase in complexity or the

[6] `http://www.lcd.upenn.edu`.

[7] ABC, CNN, VOA, NYT, PRI, and APW.

value of k. Results from both document collections indicate that as more and more topics or document clusters are added to the dataset being clustered by GD-CLS, the accuracy of the clustering decreases. For the Reuters collection, in case of $k = 2$, i.e., when dealing with only two topics, the algorithm performs with above 99% accuracy, but in case of $k = 20$, the accuracy drops down to just above 54% (Table 4). However, in case of TDT2, the drop in accuracy is much less precipitous than for Reuters (Table 5). For TDT2, for $k = 20$, accuracy is just above 80%, which seems like a significant improvement from 54% for Reuters. This disparity can be attributed to the differences in content of the two collections. Documents in the Reuters collection are categorized under broad topics (such as "earn," "interest," "cocoa," "potato," etc., listed in [13]), while for TDT2, the topic labels are much more specific (e.g., "The Asian economic crisis," "Tornado in Florida," and "Oprah lawsuit"). The very specificity of the topics in the TDT2 guarantees a heterogeneity in the document collection that is not present in the Reuters collection. In the case of Reuters, while "potato" and "zinc" may constitute very distinct clusters, "interest" and "money-fixes" do not. In fact, as noted by Xu et al. [15], there is a degree of overlapping of content across topics in the Reuters collection that contributes to the much more rapid decline of accuracy in case of Reuters than it does for TDT2.

Another notable trend that also points to the sensitivity of the GD-CLS algorithm for NMF to the contents of the document collections is the differences in accuracy for the different λ values. In case of TDT2, the different λ values for each k do not affect the performance by any noticeable amount. But for Reuters, the drop in accuracy for increasing values of the λ parameter suggests that text collections that are somewhat homogeneous in content, are more sensitive to the changes of the λ parameter (or the sparsity of the H matrix). The primary reason for using a larger λ value (or an increase in the sparsity of H) is to achieve faster computation times. Inspection of the results from Table 4 and 5 suggests that that is indeed the case, especially in higher complexity problems (Table 6).

It can be inferred from Table 6 that an increase in the sparsity of H results in a significant increase in computational speed and this holds for both TDT2 and Reuters. As for accuracy, the λ values do affect performance for Reuters but not for TDT2. However, when compared to the gain in computational time, the 2 to 3% decrease in accuracy can be considered a very reasonable tradeoff.

An aspect of GD-CLS that cannot be directly observed from the result tables is the change in performance of the factorization with regards to disparate cluster sizes. When creating document subsets for each value of k from the preclassified clusters of the Reuters or TDT2 corpus, attention is given to keep the cluster sizes within a reasonable bound of one another. This constraint,

Table 4
Results for Reuters (AC = Accuracy measure defined in Section 5.1)

k	λ	AC	CPU time (sec)
2	0.100	0.962256	2.63
2	0.010	0.963440	2.76
2	0.001	0.962262	3.19
4	0.100	0.758630	3.86
4	0.010	0.774503	4.43
4	0.001	0.777460	5.51
6	0.100	0.716229	6.51
6	0.010	0.722549	8.01
6	0.001	0.726186	10.54
8	0.100	0.572499	9.73
8	0.010	0.555926	12.79
8	0.001	0.560444	18.39
10	0.100	0.657349	30.65
10	0.010	0.673601	36.79
10	0.001	0.666243	47.75
15	0.100	0.609148	56.53
15	0.010	0.613033	74.89
15	0.001	0.618249	104.18
20	0.100	0.545806	57.26
20	0.010	0.567711	87.77
20	0.001	0.571387	122.13

which is not imposed by Xu et al. in [15], is enforced due to results obtained from experiments similar to those described in Table 7.

The imbalance in the cluster sizes in $dataset_1$ has a definite effect on the performance of GD-CLS regardless of the document corpus being used. In case of the original clusters from $dataset_1$, the ratio of $cluster_1$ to $cluster_2$ is approximately 48:1, while the clusters produced by GD-CLS have a ratio of 3:1. This implies GD-CLS performs much better on datasets that have balanced cluster sizes, such as $dataset_2$, where clustering is performed with almost 100% accuracy.

Table 5
Results for TDT2 (AC = Accuracy measure defined in Section 5.1)

k	λ	AC	CPU time (sec)
2	0.100	0.993629	2.93
2	0.010	0.993629	2.94
2	0.001	0.978329	3.00
4	0.100	0.906264	9.42
4	0.010	0.908873	9.48
4	0.001	0.925784	10.04
6	0.100	0.878919	23.38
6	0.010	0.858782	23.60
6	0.001	0.860544	25.81
8	0.100	0.858497	46.86
8	0.010	0.859123	47.42
8	0.001	0.853479	52.48
10	0.100	0.840443	97.39
10	0.010	0.836955	98.34
10	0.001	0.847155	110.26
15	0.100	0.869069	135.66
15	0.010	0.872499	140.08
15	0.001	0.870932	172.06
20	0.100	0.832097	303.54
20	0.010	0.835903	315.64
20	0.001	0.840977	405.16

Table 6
CPU time for $k = 15$ for different λ values

λ	Reuters	TDT2
0.1	56.5433	135.6620
0.01	66.0033	140.0820
0.001	93.3900	172.0670

7 Conclusions and Future Work

This study demonstrates how GD-CLS, a hybrid NMF algorithm, can be effectively used to classify text collections in an unsupervised or automated manner. The proposed algorithm can be used to construct a parts-based representation of the text data, in which the localization of the parts or features

Table 7
A comparison of results with different cluster sizes

Corpus	Dataset	Cluster	Original cluster sizes	GD-CLS generated cluster sizes
Reuters	$dataset_1$	$cluster_1$	2125	1690
		$cluster_2$	45	480
	$dataset_2$	$cluster_1$	114	112
		$cluster_2$	99	101
TDT2	$dataset_1$	$cluster_1$	1476	1231
		$cluster_2$	31	276
	$dataset_2$	$cluster_1$	110	109
		$cluster_2$	120	121

can be regularized to create a balance between computational cost and accuracy.

In its current stage, the GD-CLS algorithm for NMF is not equipped to handle updating in an efficient manner. Once the document collection has been clustered via NMF, adding a small number of documents to the collection can be achieved by comparing each of the new documents (represented by a vector) to the basis vectors and associating the new document to the basis vector or topic to which it is the most similar. But this updating technique is not scalable and would produce poor results if used to add a large number of documents that cannot be associated with any of the basis vectors.

NMF, in general, has mostly been applied to image analysis and text mining. Another field that could benefit from this technique is bioinformatics. Problems such as identifying motifs or significant features in protein sequences (partial strings of DNA) are natural candidates for application of NMF. In such problems, protein sequences can be viewed as analogous to text documents and the basis vectors or topics to motifs that control gene expression [14].

Although the primary function of NMF would be for classification as opposed to query-based information retrieval, the resulting clusters can be used to provide retrieval capabilities. Much in the style of limited updating discussed earlier, a user query can be represented by a term vector, which is then used to compute a similarity measure (e.g., cosine measurement) between the query

and the basis vectors. The basis vector or topic that yields the highest value is deemed the most relevant and documents belonging to that topic is provided to the user.

Acknowledgement

The authors would like to thank Murray Browne for his technical assistance with the production of this manuscript.

References

[1] M.W. Berry, Z. Drmač, and E. Jessup. Matrices, Vector Spaces, and Information Retrieval. *SIAM Review*, 41:335–362, 1999.

[2] M.W. Berry and M.Browne. *Understanding Search Engines: Mathematical Modeling and Text Retrieval*. SIAM, Philadelphia, PA, 1999.

[3] M.H. Dunham. *Data Mining: Introductory and Advanced Topics*. Prentice Hall, Upper Saddle River, NJ, 2003.

[4] J.T. Giles and and M.W. Berry L. Wo. GTP (General Text Parser) Software for Text Mining. In H. Bozdogan, editor, *Software for Text Mining, in Statistical Data Mining and Knowledge Discovery*, pages 455–471. CRC Press, Boca Raton, FL, 2003.

[5] D. Guillamet and J. Vitria. Determining a Suitable Metric when Using Non-Negative Matrix Factorization. In *Sixteenth International Conference on Pattern Recognition (ICPR'02), Vol. 2*, Quebec City, QC, Canada, 2002.

[6] P. Hoyer. Non-Negative Sparse Coding. In *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing*, Martigny, Switzerland, 2002.

[7] A. Hyvärinen and P. Hoyer. Emergence of Phase and Shift Invariant Features by Decomposition of Natural Images into Independent Feature Subspaces. *Neural Computation*, 12:1705–1720, 2000.

[8] D. Lee and H. Seung. Algorithms for Non-Negative Matrix Factorization. *Advances in Neural Information Processing Systems*, 13:556–562, 2001.

[9] S. Mironova. Integrating Network Storage into Information Retrieval Applications. Master's thesis, Department of Computer Science, University of Tennessee, Knoxville, TN, 2003.

[10] Z. Mu, R. Plemmons, and P. Santago. Iterative Ultrasonic Signal and Image Deconvolution for Estimating the Complex Medium Response. In *IEEE Transactions on Ultrasonics and Frequency Control*. IEEE, 2003. Submitted for publication.

[11] V. Pauca, F. Shahnaz, M.W. Berry, and R. Plemmons. Text Mining Using Non-Negative Matrix Factorizations. In *Proceedings of the Fourth SIAM International Conference on Data Mining*, Lake Buena Vista, FL, April 22-24, 2004. SIAM.

[12] S. Prasad, T. Torgersen, V. Pauca, R. Plemmons, and J. van der Gracht. Restoring Images with Space Variant Blur via Pupil Phase Engineering. *Optics in Info. Systems, Special Issue on Comp. Imaging, SPIE Int. Tech. Group Newsletter*, 14(2):4–5, 2003.

[13] F. Shahnaz. Clustering Method Based on Nonnegative Matrix Factorization for Text Mining. Master's thesis, Department of Computer Science, University of Tennessee, Knoxville, TN, 2004.

[14] G.W. Stuart and M.W. Berry. Comprehensive Whole Genome Bacterial Phylogeny Using Correlated Peptide Motifs Defined in a High Dimensional Vector Space. *Journal of Bioinformatics and Computational Biology*, 1(3):475–493, 2003.

[15] W. Xu, X. Liu, and Y. Gong. Document-Clustering based on Non-Negative Matrix Factorization. In *Proceedings of SIGIR'03, July 28-August 1*, pages 267–273, Toronto, CA, 2003.