Algorithm Comparison for Document Clustering and Topic Modeling

Andrew Roberts
5/11/2018

1. Problem Statement

From presidential speech transcripts to news articles, the world is full of bodies of documents that may reveal valuable insights when analyzed properly. Especially useful is the ability to identify the core topics in a set of documents and then cluster subsequent documents according to this topic identification. In this paper I compare the performance of the Non-Negative Matrix Factorization with that of other algorithms in accomplishing these tasks.

2. Introduction

With the rising level of interest in processing large corpora, the research community has continually expanded the Natural Language Processing (NLP) tool-kit utilized in a variety of human language applications. Topic modeling and document clustering are two such applications that have received significant attention in the NLP field. A relatively recent class of algorithms, falling under the umbrella term *Non-Negative Matrix Factorization* (NMF), possesses the functionality to effectively address both of these fundamental NLP tasks. In this paper, I provide background on the application of NMF to these tasks, as well as commonly used alternative models. I then apply the algorithms to a publicly available, preprocessed corpus and compare the results. This paper does not constitute a comprehensive analysis of these algorithms; rather, it seeks to present an intuitive implementation and application of common NLP tools, as well as demonstrate how the results of each algorithm may differ on a specific dataset.

3. Background

3.1. Non-Negative Matrix Factorization (NMF)

NMF is a matrix factorization technique that seeks to represent a non-negative matrix, **X**, as the approximate product of two non-negative matrices, **W** and **H**:

$$\boldsymbol{X}_{mxn} \approx \boldsymbol{W}_{mxr}\boldsymbol{H}_{rxn}$$

where $r < \min(m, n)$.[i] If **X** is a high-dimensional matrix and r is considerably smaller than both m and n, then we can think of **X** as the product of a tall, skinny matrix and a short, wide matrix. Rewriting the above equation as

$$\boldsymbol{X}_{mxn} \approx \boldsymbol{W}_{mxr}\boldsymbol{H}_{rxn} = [\boldsymbol{W}\boldsymbol{h}_1\ \boldsymbol{W}\boldsymbol{h}_2 \ldots \boldsymbol{W}\boldsymbol{h}_n]$$

where $\boldsymbol{h}_j$ is the j^{th} column of **H**, then it becomes clear that **W** can be thought of as containing a basis optimized for the linear approximation of the data in **X**.[ii] Each column of **X** is represented as a linear combination of the columns of **W**, weighted by the entries of the corresponding column of **H**. Because these matrices contain only non-negative entries, this linear combination view of NMF sheds light on its natural clustering property. Each column of **X** is expressed as an additive combination of the columns of **W**, leading to a "parts of a whole" representation of the data.[iii]

It turns out that this factorization has no closed-form analytical solution. Instead, the factors are approximated iteratively by minimizing some distance function $D(\boldsymbol{X} \mid \boldsymbol{WH})$ that quantifies the error between **X** and **WH**. While many different choices for D are available, for this application I choose the simplest option, where D is the Frobenius Norm $\|\boldsymbol{X} - \boldsymbol{WH}\|_F^2$. This function can be minimized utilizing a block coordinate descent approach, which entails performing gradient descent on each dimension separately, while holding the other constant. This leads to the following optimization problem:

$$\{\boldsymbol{W}^*, \boldsymbol{H}^*\} = argmin\ \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{WH}\|_F^2\ ; \boldsymbol{W}, \boldsymbol{H} \geq 0$$

Minimizing this function by employing the described technique, the following multiplicative update rules are obtained.

$$\boldsymbol{W}_{ij} \leftarrow \boldsymbol{W}_{ij} \frac{(\boldsymbol{XH}^T)_{ij}}{(\boldsymbol{WHH}^T)_{ij}}$$

$$\boldsymbol{H}_{ij} \leftarrow \boldsymbol{H}_{ij} \frac{(\boldsymbol{W}^T\boldsymbol{X})_{ij}}{(\boldsymbol{W}^T\boldsymbol{WH})_{ij}}$$

3.2. Singular Value Decomposition (SVD)

The SVD of a real-valued matrix **X** decomposes it into a product of an orthonormal matrix, a rectangular diagonal matrix, and a second orthonormal matrix.

$$\boldsymbol{X}_{mxn} = \boldsymbol{U}_{mxm}\boldsymbol{S}_{mxn}\boldsymbol{V}^T_{nxn}$$

The diagonal entries of **S** are called the *singular values*, and are simply the square roots of the eigenvalues of $\mathbf{X^TX}$. The columns of **U**, or left-singular vectors of **X**, are the eigenvectors of $\mathbf{XX^T}$. Similarly, the columns of **V**, or right-singular vectors of **X** are the eigenvectors of $\mathbf{X^TX}$. Specific SVD algorithms are beyond the scope of this paper; rather, its primary function here is in its ability to obtain a low-rank approximation of the original matrix.

4. Methods

4.1. Document Representation

Let $\mathbf{X_{raw}}$ be an *m x n* matrix whose entries correspond to counts of terms in documents, where each row represents a unique term and each column a unique document. Concretely, the row representing term *t* and the column representing document *d* has entry $f_{t,d}$, the number of times *t* appears in *d*. I convert this raw data matrix into the final term-document matrix **X**, whose entries are calculated using the following tf-idf equations:

$$\boldsymbol{X}_{ij} = tfidf(t_i, d_j) = tf(t_i, d_j) * idf(t_i, D)$$

$$tf(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t'd}}$$

$$idf(t,D) = log\left[\frac{1+n}{1+df(d,t)}\right] + 1$$

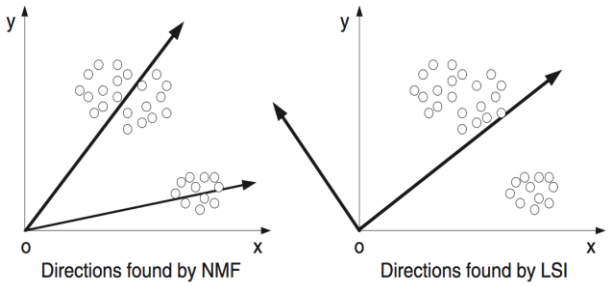
where *df(d, t)* is the number of documents that contain term *t* (i.e., the document frequency). Note that the *i, j* subscripts were dropped in the second two equations for simplicity. The addition of one to the numerator and denominator inside the logarithm of the *idf* equation is to avoid divide-by-zero errors (if a term does not appear in any documents). The addition of one at the end of the equation ensures that words appearing in all documents are not completely ignored.

4.2. Topic Modeling

Topic modeling is a common NLP task that seeks to extract abstract base topics from a set of documents. Topic modeling algorithms rest on the assumption that there exists latent, semantic structures underlying the observed data. This suggests that each document vector can be represented as a linear combination of some smaller number of "base topic" vectors. In other words, the base topics correspond to the principal axes in the document space, thus forming a lower-dimensional basis that allows for a more efficient representation of the original documents.

4.2.1 Topic Modeling with NMF

Applying NMF directly to the term-document matrix **X** yields the factors **W** and **H.** As mentioned previously, **W** can be thought of as containing a basis optimized for the linear approximation of the original data. Therefore, one can think of each column of **W** as an abstract base topic existing in the document space. In this context, the non-negativity constraints are quite convenient as the columns of **W** can be normalized such that each vector defines a probability distribution over the words in each base topic. By viewing the most probable words in each vector one can attempt to determine the corresponding topics. The base topic vectors found via NMF are not subject to an orthogonality constraint, allowing for the overlap of words across the vectors. Intuitively, this seems reasonable as even very disparate topics are expected to share some words.



4.2.2 Topic Modeling with SVD

Applying SVD directly to the term-document matrix **X** yields the factors **U, S**, and $\mathbf{V^T}$. To reduce the dimensionality of the dataset, and thus uncover a lower-dimensional basis, the first *k* columns of **U**, upper *k x k* portion of **S,** and first *k* rows of $\mathbf{V^T}$ are sliced from the original matrices. Therefore, $\boldsymbol{X}_k = \boldsymbol{U}_k \boldsymbol{S}_k \boldsymbol{V}_k^T$ represents a rank *k* approximation of the data. This process is equivalent to Principal Components Analysis (PCA) when **X** is centered (i.e., the column-wise means are 0). In LSA, SVD is usually applied to the un-centered term-document matrix in order to maintain its sparsity. Of particular interest in this case is the reduced matrix $\mathbf{U_k}$, whose columns form the lower-dimensional basis of interest. In contrast to NMF, these base topic vectors are subject to an orthogonality constraint. Additionally, interpreting the semantic topics is made more difficult by the potential presence of negative entries in **U**; these can be interpreted as the *absence* of words in a topic. While various methods have been proposed to deal with this, I choose to look at two distinct options: 1.) rank by absolute values, and 2.) rank as one would on a normal number line. The first approach makes the implicit assumption that latent topics are defined not only by the presence of words, but also by their absence. An important consequence of the presence of negative entries is that it allows the same words to appear with non-zero weights across various topic vectors, so long as the negatives and positives cancel out to ensure orthogonality. This sheds more light on the difficulty of interpreting a basis with negative values in the document space.

4.3. Clustering

As with topic modeling, clustering algorithms view each document as a point existing in m-dimensional space. Given an integer *k,* one seeks to find *k* distinct "groups" of documents, with each group ideally sharing common features. This task is somewhat similar to topic modeling, but does not seek to uncover a low-dimensional basis in the document space. Rather, the goal is simply to discover clusters of similar documents, where each cluster may or may not have intuitive meaning.

4.3.1 Clustering with NMF

As mentioned previously, the base topics uncovered via NMF can be viewed as existing in $\mathbb{R}^m$, the m-dimensional document space. Because the original documents also exist in this space, a document can be assigned to the topic vector that minimizes the angle between the two vectors. As covariance is closely related to the angle between vectors, I simply assign each document to the topic vector that maximizes the document-topic correlation.

4.3.2 Clustering with K-Means

Again considering each document as a point in $\mathbb{R}^m$, k-means seeks to partition the documents in *k* clusters (in this case, *k* = 5). The algorithm adopts a Vector Quantization (VQ) approach in which each cluster is defined by its respective *centroid,* defined simply as the mean of all observations in the cluster. Given clusters

$C_1, C_2, \ldots, C_k$ and their respective centroids $c_1, c_2, \ldots, c_k$, k-means attempts to minimize the intra-cluster sum of squared errors (SSE):

$$SSE = \sum_{i=1}^{k} \sum_{d_j \in C_i} \|d_j - c_i\|^2$$

where $d_j \in \mathbb{R}^m$ denotes a document in the corpus. It addresses this task using an iterative approach, continuously recalculating cluster centroids and re-assigning documents to the new centroids (based on Euclidean distance) until the SSE no longer decreases.

5. Data

All algorithms are performed on a corpus consisting of 2225 articles from the BBC news website, each corresponding to one of five general categories: business, entertainment, politics, sport, and technology. All articles were published in 2004 or 2005. Conveniently, the data has already been preprocessed and the resulting term-document matrix is available for download. The dataset has been stemmed (Porter algorithm) and has undergone both stop-work removal and low-term frequency filtering (count < 3). In its downloadable form, the matrix entries represent word counts, but I alter them according to the following tf-idf equations presented in Section 3.1.

6. Evaluation

6.1. Low-Rank Approximation

To quantify the quality of the low-rank approximation uncovered by either NMF or SVD, I calculate the relative Frobenius Norm:

$$Approximation\ Error = \frac{\|\boldsymbol{X} - \boldsymbol{X}_{approx}\|_F^2}{\|\boldsymbol{X}\|_F^2}$$

where $\boldsymbol{X}_{approx} = \boldsymbol{U}_k \boldsymbol{S}_k \boldsymbol{V}_k^T$ for SVD and $\boldsymbol{X}_{approx} = \boldsymbol{W}\boldsymbol{H}$ for NMF.

6.2. Clustering

For this analysis I quantify the quality of the clustering as the proportion of correct classifications over the total number of observations:

$$Clustering\ Accuracy = \frac{\sum_i 1\{predicted\ label = actual\ label\}}{n}$$

For a paper more focused on clustering evaluation, additional metrics such as precision and recall would typically be utilized, but for the purposes of this analysis this basic evaluation criterion serves well to emphasize the performance differences of the two algorithms.

7. Results

7.1. Topic Modeling

Utilizing the strategies discussed above, I examine the most prominent terms present in the base topic vectors discovered by both NMF and SVD. Determining topics is somewhat arbitrary because topics are assigned simply by looking at the terms associated with the largest entries in each basis vector. However, for NMF the interpretation is made far easier due to the non-negativity constraints. I run NMF on **X** and report the top ten most probable terms in each normalized column of **W** below.

Politics	Technology	Sports	Business	Entertainment
labour	mobil	game	bank	film
elect	phone	plai	sale	award
parti	user	win	growth	best
blair	peopl	england	compani	star
tori	technolog	player	price	actor
tax	softwar	match	market	oscar
brown	servic	club	year	nomin
govern	music	cup	share	festiv
minist	microsoft	against	firm	actress
howard	comput	injuri	profit	music

Here, the i^{th} row contains the i^{th} most probable term in each vector. NMF has clearly identified the five prominent latent topics in this corpus, so I label the columns accordingly. The approximation error of the factorization (measured by the relative Frobenius Norm) is about .97.

I proceed to follow a similar approach using SVD. However, due to the presence of negative entries, the interpretation of the latent topics in this case becomes more difficult. To address this, the first strategy I employ is to list the top ten terms according to the absolute value of their entries. Thus, the latent topics will be composed of terms strongly associated with the topic as well as those whose *absence* is strongly associated with the topic. The results are shown below, ranked in decreasing order of the absolute magnitude of the terms' associated weights. Asterisks indicate negative weights.

?	Politics	Sports	?	Technology
year*	film*	film*	labour*	phone
film*	award*	award*	elect*	mobil
game*	best*	england*	parti*	user
peopl*	plai*	music*	film*	bank*
plai*	game*	best*	blair*	peopl
on*	star*	win	tori*	softwar

compani*	elect	game	game	microsoft
firm*	win*	plai	brown*	program
world*	labour	match	tax*	growth*
govern*	govern	club	award*	economi*

Clearly, identifying latent topics is not as straightforward as with the NMF case. While some topics are still identifiable, those characterized primarily by the absence of words are subject to debate.

An alternative strategy in topic identification using SVD is to simply rank the terms according to their raw weights. Therefore, negative weights will only appear in the top ten rankings if a topic is composed nearly entirely of negative-weighted terms.

?	Politics	Sports	Business	Technology
utter*	elect	england	game	phone
prai*	labour	win	firm	mobil
dread*	govern	game	sale	user
195*	tax	plai	mobil	peopl
smack*	parti	match	phone	softwar
selfcontain*	firm	club	compani	microsoft
remiss*	bank	cup	market	program
constru*	economi	labour	player	comput
eyeopen*	blair	injuri	bank	technolog
indecis*	tori	elect	profit	search

This second strategy appears to achieve slightly more interpretable results on this dataset. However, it identifies one topic composed entirely of negative-weighted terms. The SVD approximation error is about .969, slightly lower than the NMF case. This is unsurprising as it can be shown that the SVD factorization minimizes the Frobenius norm between $\mathbf{X}$ and $\mathbf{X}_{\mathbf{approx}}$.

7.2. Clustering

Following from section 7.1, I obtain the five columns of $\mathbf{W}$ and proceed to utilize the clustering strategy discussed in section 4.3.1. This method obtains a clustering accuracy of 84.58%. The results broken down by document class are given below.

Document Class	Clustering Accuracy
Business	96.1%
Sports	88.1%
Politics	87.8%
Tech	78.3%
Entertainment	67.9%

All	84.6%

These findings indicate that business related documents may contain a more distinctive core lexicon, while articles related to entertainment tend to use more diverse language.

Next, I perform a k-means clustering analysis on the term-document matrix. It should be noted that the clustering algorithm is run on the entire matrix, rather than a reduced rank approximation. Using k-means, the clustering accuracy drops significantly to 54.2%. However, as demonstrated by the table below, the interpretations of business and entertainment articles are further supported.

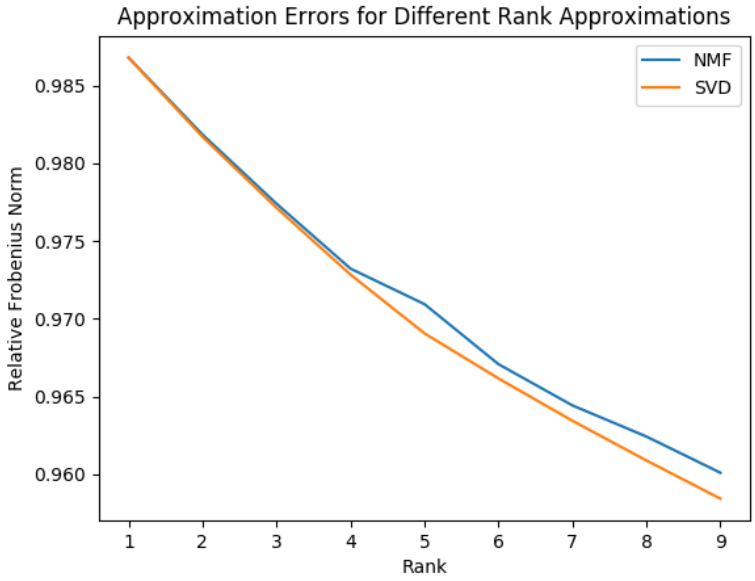
Document Class	Clustering Accuracy
Business	100.0%
Tech	92.4%
Sports	43.8%
Entertainment	13.7%
Politics	0.0%
All	54.2%

Although a thorough discussion of the mathematical limitations of k-means are beyond the scope of this paper, the relative positions of the vectors in the document space may be contributing to these sub-optimal results. Alternative techniques such as Spectral Clustering are often used in NLP settings to address this issue. Furthermore, the fact that the NMF base topic vectors were constructed from the same dataset on which the clustering was performed may be inflating the accuracy achieved using this technique. Despite these caveats, this simple experiment demonstrates the effectiveness of NMF as a clustering algorithm and supports the results found in recent literature.[iv]

8. Conclusion

This paper explores the application of NMF to two distinct NLP tasks: topic modeling and document clustering. I compare the low-rank approximations of the term-document matrix achieved by NMF and SVD, concluding that the non-negativity constraints present in the NMF update rules result in relatively more easily interpreted topic vectors. I then proceed to utilize these non-negative base topic vectors to cluster the articles in the corpus, comparing the clustering accuracy to that achieved by k-means. Although the NMF results may be inflated due to the fact that testing was performed on the same term-document matrix, I find that NMF outperforms k-means, consistent with results found in recent literature.[v]

Appendix



References

[i] Daniel D. Lee and Sebastian H. Seung, "Algorithms for Non-Negative Matrix Factorization," *MIT Press*, Neural Information Processing Systems, 2000, 556–62.
[ii] Lee and Seung.
[iii] Lee and Seung.
[iv] Wei Xu, Xin Liu, and Yihong Gong, "Document Clustering Based On Non-Negative Matrix Factorization," *Special Interest Group on Information Retrieval*, 2001, ccrl.sj.nec.com.
[v] Xu, Liu, and Gong.