Interactive Document Clustering

Baalbaki, Nabil
nbaalbaki3@gatech.edu

Meek, Christopher
cmeek3@gatech.edu

Saunders, Jonathon
jsaunders36@gatech.edu

Shields, Nicholas
nshields9@gatech.edu

Ziai, Amir*
aziai3@gatech.edu

Georgia Institute of Technology - CSE 6242 Fall 2018

Abstract

Document similarity is used widely today in web searches, libraries, and throughout the corporate world. It is a critical aspect of the increasing dependency on and analysis of data in todays climate. While important, past approaches have their drawbacks, and we hope to improve on them. Our team has implemented an interactive clustering visualization tool for exploratory data analysis of text datasets. This tool implements consistent document similarity techniques and clustering methods to allow the user to experiment with visualizing their data.

Keywords: UI; Visualization; Document Similarity; Document Clustering; K-means; Interactive

1 Introduction

Document similarity is used widely today in web searches, libraries, and throughout the corporate world. It is a critical aspect of the increasing dependency on and analysis of data in todays climate. While important, past approaches have their drawbacks, and we hope to improve on them. Our team has implemented an interactive clustering visualization tool for exploratory data analysis of text datasets. This tool implements consistent document similarity techniques and clustering methods to allow the user to experiment with visualizing their data.

Current approaches to clustering without user feedback are inaccurate or are too unstable. In the k-means approach, a ground truth is usually required to determine k [14] while methods like LDA are stochastic, meaning the output can change dramatically with each run [12]. Our approach attempts to hand control to the user in digestible pieces throughout the clustering process, through interaction with the graph, and, for power users, the choice of the algorithm. This allows for a better balance of background tuning and user feedback, producing more consistent results, and providing a better user experience.

*All authors contributed equally to this work.

The ability to quickly, reliably, and effectively move between connected content is valuable across many disciplines. Improving our approach to document similarity enables faster research and learning as potentially sparse data becomes easier and more enjoyable to navigate and visualize.

2 Method

One of the primary focuses in development of the Interactive Document Clustering tool was to make the tool as intuitive as possible, especially for non-data-oriented users. This tool allows users to specify the data set (from a pre-processed list including "BBC", "20 News Groups", and "All the News"), the algorithm (iKMeans, Birch, Spectral Clustering, and Affinity Propagation), and the number of clusters if the algorithm requires it. Once the specified clustering is performed an interactive chart is provided as well as a table of terms per cluster. This display allows the user to review the clustering method by evaluating the terms in each cluster. If desired the user can edit the terms per cluster, i.e., edit, merge, or split the terms per cluster, and visualize the change in the cluster visualization. Alternatively the user can manipulate the number of clusters or the clustering algorithm to find the best representation of the data.

Prior to computing document similarity/clustering the text data has to be represented as vectors. Before implemeting the selected algorithm the tool uses a CountVectorizer, i.e., counts the frequency of a word in each document, or a TFIDFVectorizer, which is similar but is normalized over the corpus.

The Interactive Document Clustering tool is implemented primarly in Python 3.7 and Javascript. The clustering methods chosen for visualization were selected based on a review of the documentation for the scikit-learn Python framework [16]. The default method is k-means with a value of $k = 5$. Users can manipulate these values if they choose. These input values are then used to seed the k-means algorithm for the next round which has proven very successful in consistently producing clusters that are aligned with the user input. In addition to k-means the tool can also use BIRCH, Spectral Clustering, and Affinity Propagation methods.

2.1 KMmeans

KMeans chooses k centroids randomly and then iterates over the following two steps until convergence. [16]

1. The first step is an assignment step in which the distance between each point, x, and each centroid, c_i is calculated for all $c_i \in C$ where C is the set of centroids and x is each point in the dataset:

$$\underset{c_i \in C}{\operatorname{argmin}} ||x - c_i||^2 \tag{1}$$

2. The second step is a centroid update step in which the centroids are recalculated based on the updated assignments in step 1:

$$c_i = \frac{1}{|S_i|)} \sum_{x_i \in S_i} x_i \tag{2}$$

2.2 Birch

BIRCH stands for "balanced iterative reducing and clustering using hierarchies." Birch is often used with multi-dimensional datasets because of its ability to quickly cluster them with minimal processing time using the following steps and equations: [16]

1. Load the data and build a CF tree - $CF = (N, LS, SS)$.
2. Create initial clusters.
3. Refine the clusters and remove outliers if desired.

$$N = \text{number of data points} \tag{3}$$

$$LS = \sum_{i=1}^{N} x_i \tag{4}$$

$$SS = \sum_{i=1}^{N} (x_i)^2 \tag{5}$$

$$\text{centroid } c_i = \frac{LS}{N} \tag{6}$$

$$\text{avg. dist from a point in a cluster to its centroid } r_i = \sqrt{\frac{\sum_{i=1}^{N}(x_i - c_i)^2}{N}} \tag{7}$$

$$\text{avg. dist. in a cluster } d_i = \sqrt{\frac{\sum_{i=1}^{N}\sum_{j=1}^{N}(x_i - x_j)^2}{N(N-1)}} \tag{8}$$

2.3 Spectral Clustering

Spectral Clustering used a linear algebra based algorithm to compute KMeans over eigenvalue vectors using the following steps: [16]

1. Generate similarity matrix.
2. Compute eigenvectors.
3. Perform k-means over eigenvectors.

2.4 Affinity Propagation

The Affinity Propagation method determines how well a given point, an exemplar, explains another point. In terms of previous methods an exemplar is similar to a centroid. This method is the only method this tool uses where the user does not need to specify a ground truth. The algorithm is as follows: [16]

1. Compute the similarity between points:

$$s(i, k) = -||x_i - x_k||^2 \tag{9}$$

2. Compute a responsibility matrix "r", i.e., a matrix demonstrating the suitability of a point k as an exemplar for i, and an availability matrix "a", i.e., a matrix for the suitability of a point i to chose k. These two matrices are updated for each point by iterating over the following definitions until the clusters are no longer changing:

$$r(i,k) \leftarrow s(i,k) - \max_{k' \neq k}\{a(i,k') + s(i,k')\} \tag{10}$$

$$a(i,k) \leftarrow \min_{i \neq k}\left(0, r(k,k) + \sum_{i' \notin \{i,k\}} max(0, r(i',k))\right) \tag{11}$$

$$a(k,k) \leftarrow \sum_{i' \neq k} max(0, r(i',k)) \tag{12}$$

Results from the chosen algorithm are displayed using Highcharts and further details can be accessed by hovering over any data point. A word cloud is generated per cluster by hovering over a given cluster's dictionary.

3 Data Sources

The current prototype provides three data sets for user analysis: "BBC", "20 News Groups", and "All the News."

Table 1: Descriptive Statistics of Data Sources

Dataset	Data Source	# of Observations	Size on Disk (MB)
BBC	Kaggle	2,225	5
20 News Groups	Ken Lang [18]	11,314	95
All the News	Kaggle	50,006	194

4 Using the Tool

The Interactive Document Clustering tool can be used to explore the three datasets listed in section 3 using the algorithms described in section 2. By default the tool is set to the BBC using the KMeans algorithm with $k = 5$. The results can be visualized in the tool as seen in Figure 1. In addition k clusters are shown with their Silhouette score, and a subset of key terms closest to each centroid.

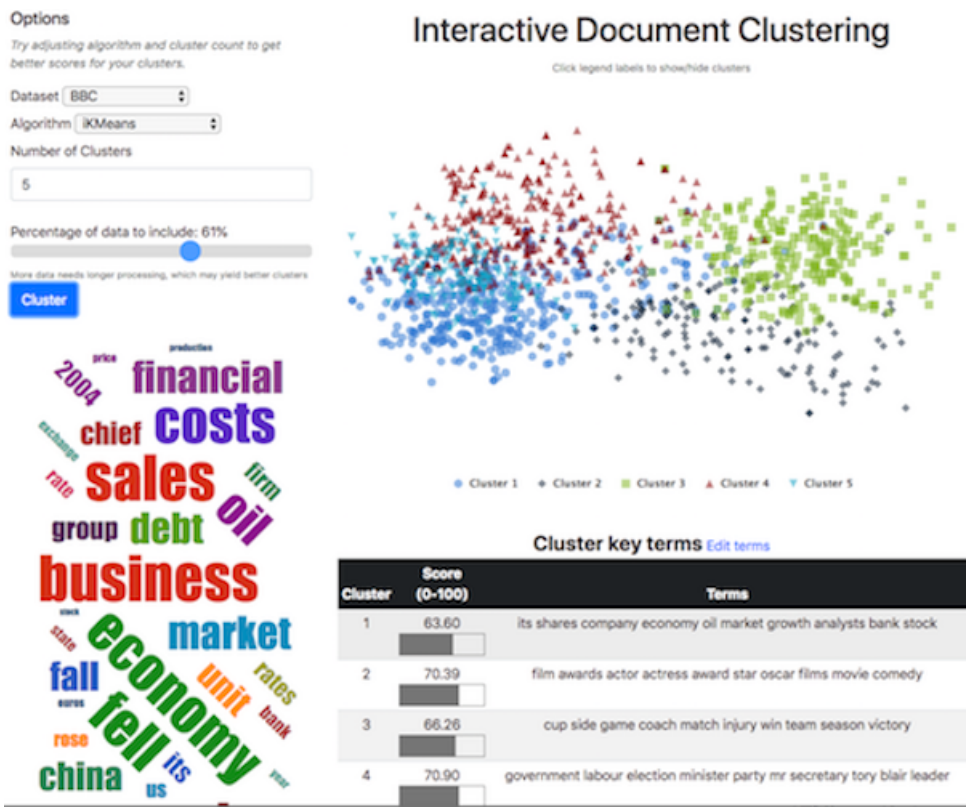

Figure 1: Default UI for Interactive Document Clustering tool.

Upon reviewing the results the user has the ability to edit the terms assigned to each centroid in order to optimize the score or rerun the algorithm with a different number of clusters.

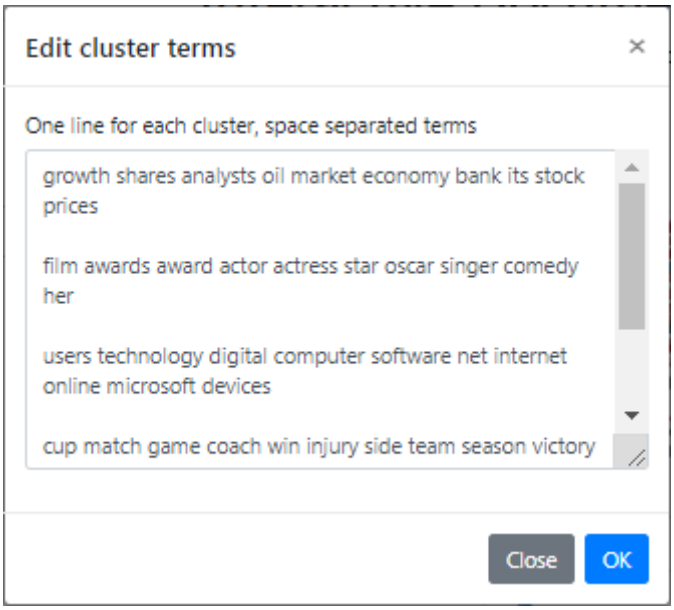

Figure 2: The user can edit the values in each cluster to optimize the next run.

5 Evaluation

We started by designing an experiment to evaluate the performance of different clustering algorithms across a number of dimensions and to answer the question of whether or not an interactive document clustering tool is more valuable (accuracy, time, and quality of results) than a static tool. In these experiments we used the "BBC" and "20 news groups" datasets which are categorized into five and twenty groups, respectively. The BBC is relatively straightforward to cluster. There are only five categories and the articles are distinct enough. The 20 news groups dataset has many categories and the boundaries between them are very fuzzy.

We used the Adjusted Mutual Information (AMI) score for evaluating the quality of different clustering algorithms. AMI is calculated as follows:

$$AMI(U,V) = \frac{MI(U,V) - E(MI(U,V))}{max(H(U)H(V)) - E(MI(U,V))}$$

where $MI(U,V)$ is the mutual information between U and V and $H(U)$ is the entropy for U.

For our evaluation V is the labeling produced by a clustering algorithm and U is the actual labels. The AMI score is upper-bounded at 1 when the clustering is perfect. For instance if U=['entertainment', 'sport', 'sport'] and V=[0, 1, 1] then $AMI(U,V) = 1$ as the algorithm has correctly clustered the articles.

In addition we considered the effects of the number of chosen clusters on quality. All of the algorithms we have considered in this project, with the exception of Affinity Propagation, take the number of desired clusters as a hyperparameter. Finally we considered the effects of the number of clusters and the choice of algorithm on clustering time.

The following table summarizes our results sorted by AMI:

Table 2: Comparing clustering quality and wall time

Dataset	Algo	AMI	Wall time (s)
BBC	KMeans	0.76	11.7
BBC	Spectral	0.65	0.5
BBC	Birch	0.52	2.0
20 News Groups	Kmeans	0.21	3.4
BBC	Affinity	0.21	11.4
20 News Groups	Spectral	0.20	0.8
20 News Groups	Birch	0.17	2.3
20 News Groups	Affinity	0.01	7.3

As evident from our results KMeans is performing relatively well for these datasets. The next set of our results emphasize the importance of selecting the correct number of clusters (k):

The results of these experiments suggested that we need to allow the user to iteratively provide feedback to the algorithm. The feedback process should be very intuitive and must be highly informative to the algorithm to optimize the algorithm in as little iterations and time as possible. We modified the algorithm to make it iterative by soliciting "key terms" from the user that define each cluster. Initially the algorithm runs with a default or a user-defined number of clusters. After this initial run we present a subset of terms closest to each centroid to the user as key terms. Given this initial seed the user can make modifications and re-run the algorithm. The provided key terms are used to initialize the new centroids.

Table 3: Comparing clustering quality and wall time for different values of k

k	AMI (BBC)	AMI (20 News)	Wall time (BBC)	Wall time (20 News)
2	0.31	0.06	9.9	4.2
5	0.72	0.13	11.6	4.7
10	0.53	0.18	14.5	4.1
15	0.50	0.21	12.9	3.7
20	0.46	0.19	19.6	5.7

For instance the following were collected for the BBC dataset from a user during user testing:

- government, tony, blair, party, liberal
- film, award, actor, actress, singer, oscar
- game, match, cup, victory, football, soccer, season
- economy, growth, oil, shares, demand, analyst
- digital, internet, technology, phone

This initialization results in an intuitive clustering with an average AMI of 0.77 across multiple runs.

Based on these experiments we designed an interactive user interface for allowing the user to cluster one of the three datasets using the algorithms we have discussed. We conducted user testing with 20 potential users. Each testing session consisted of an initial introduction to the tool accompanied by a demo of the different choices. After this initial orientation we asked the user to find the best clustering for each of the two datasets. Half of the participants were assigned to a version of the UI that included the interactive k-means and the other half were assigned to a version with static k-means.

The following summarizes the results from these user testings:

Table 4: User testing results

Dataset	Iterations w/interactive k-means	Iteration w/static k-means
BBC	1.5 ± 0.5	2.2 ± 0.9
20 News Groups	4.6 ± 2.0	7.3 ± 2.3

6 Resources

6.1 Cost

An EC2 instance (t2.medium) was used for hosting the tool at $0.05 per hour for 37 days for a total of $44.40.

6.2 Project Timeline

The Interactive Document Clustering tool was developed on the following schedule:

Table 5: Report of Activities

Project Timeline	Tasks	Dates	Hours	Status
Sprint 0	Research and Proposal	9/26/2018 to 10/10/2018	100	Done
Sprint 1	Lexical clustering and static cluster visualization	10/11/2018 to 10/24/2018	100	Done
Sprint 2	Add semantic clustering and at least 2 interactive parameters for visualization and Progress Report	10/25/2018 to 11/7/2018	100	Done
Sprint 3	Improve current UI with more interactive algorithms	11/8/2018 to 11/21/2018	100	Done
Sprint 4	Final Report and Poster	11/22/2018 to 11/28/2018	75	Done

7 Related Work

7.1 Document Analysis & Clustering

Computing document similarity is often done by representing documents as vectors and subsequently applying mathematical functions. Term Frequency, Inverse Document Frequency (TF-IDF) is a standard method of converting documents to vectors by computing the frequency of each word relative to its frequency in the entire dataset [7]. This produces a lexical representation similar to what is done in Patil, et al.s study of WordNet [10]. Other approaches can compute a semantic representation which implements a higher level of abstraction, representing the documents concepts rather than the underlying words [3]. The effectiveness of TF-IDF and other vector conversions can be increased by implementing standard NLP techniques such as stop word removal and stemming. Once a vector representation has been established one can compute similarity measures such as Euclidean distance, Jaccard coefficient, Pearson correlation coefficient, Dice Coefficient, and Kullback-Leibler Divergence (KLD). Huang compared these metrics and found KLD and Pearsons to be the most effective for a variety of corpora [8], while a previous study found the Dice Coefficient to closely mirror a human judge [5]. Choosing a good similarity measure is crucial for document clustering.

One method of clustering examines trends by implementing the single linkage algorithm, which focuses on trends-by-word frequency [11]. Another method maps the largest projection value between a document and a topic using matrix factorization [15]. However this methodology is computationally expensive. Dias et. al experiment with an information-theoretic dissimilarity measure for clustering [4].

None of the methods discussed thus far incorporate user feedback, which can be an important aspect of understanding the success of document clustering. Agrawal, et al. emphasize

this in a study that prompts the user for correlation between word pairs, which guides the merging and splitting of topics [1]. Moreover, the instability of topic modeling algorithms is well documented, and part of our design is to emphasize consistency. This instability is also seen in Latent Dirichlet Allocation (LDA) which uses a stochastic sampling process. Sherkat, et al. suggest a novel methodology that improves the stability of LDA by tuning it with Differential Evolution [12]. We correct for this in our KMeans implementation by incorporating user feedback between runs. Hu, et al. present a visualization tool that uses both Lexical Double Clustering and K-means in collaboration with t-SNE to visualize clusters and incorporate user feedback to modify clusters based on key-terms [6]. Hu's analysis is similar to Wang and Koopman who compare the Louvain method and K-means [14], but acknowledge the drawback of requiring a ground truth to proceed.

7.2 Dimensionality Reduction & Visualization

High-dimensional data needs to be mapped to two or three dimensions for more effective visualization. Van der Maaten and Hinton introduced t-Distributed Stochastic Neighbor Embedding (t-SNE), which uses a t-distribution over a Gaussian to alleviate the tendency for data points to cluster in the center of the map associated with SNE [13]. Arora et. al demonstrate the value of t-SNE in deterministic systems in comparison to Principal Component Analysis and Random Projection in a series of experiments [2]. Similarly, McAllister, et al. compared Latent Semantic Analysis (LSA), which uses Singular Value Decomposition and term-document matrices, with his own implementation of text abstraction, which instead uses a computed abstraction path to compute document classification [9]. Many comparable methods of dimensionality reduction have been developed in the last decade. These methods provide valuable insight into the importance of accurately reducing the dimensionality of the data for clearer cluster visualization.

8 Discussion and Conclusion

The current version of the Interactive Document Clustering tool allows the user to interactively explore three pre-defined datasets of news article documents. Initially the application starts with 5 clusters and the BBC dataset, and then the user has an opportunity to refine this clustering by specifying key-terms that characterize each cluster. In our limited user testing conducted with a handful of potential users we have seen very promising results.

Our success metric is how many iterations it takes the user to arrive at the correct number of clusters for this dataset. Although there is some ambiguity in this task, most users get to the same number of clusters in two iterations or less. In the background, we are calculating a number of metrics, such as the Silhouette score, that capture the quality of clustering. This is presented to the user as a way of evaluating their clustering, however, using this score can lead to over-fitting so we will explore presenting more metrics such as perplexity, adjusted mutual information, and homogeneity [17] to the user to test whether this information is useful and can further guide the clustering process. We will also expand our benchmark to include a more diverse set of datasets for evaluation purposes. We are, however, logging the consistency of the cluster quality scores for the same dataset across multiple users as a proxy for algorithm stability. Our current implementation is highly stable across runs given the same or similar set of user inputs.

Our goal was to provide the user with an interface that allows for finding consistent and high-quality insights with minimal effort, reduce the stochasticity of current models and improve stability, and allow for better user control. It is tempting to add many features and to

provide very fine-grained control to the user, but our testing suggests that this may come at the cost of user confusion and frustration.

9 Acknowledgement

This project was conducted as part of CSE 6242 in the OSMCS program at Georgia Institute of Technology under the guidance of Prof. Polo Chau. We would like to acknowledge his help and instruction in conducting the project and presenting our results.

References

[1] Agrawal, A., Fu, W., & Menzies, T. (2018). What is wrong with topic modeling? And how to fix it using search-based software engineering. Information and Software Technology, 98, 74-88.

[2] Arora, S., Hu, W., & Kothari, P. K. (2018). An Analysis of the t-SNE Algorithm for Data Visualization.

[3] Chim, H., & Deng, X. (2008). Efficient phrase-based document similarity for clustering. IEEE Transactions on Knowledge and Data Engineering, 20(9), 1217-1229.

[4] Dias, L., Gerlach, M., Scharloth, J., & Altmann, E. G. (2018). Using text analysis to quantify the similarity and evolution of scientific disciplines. Royal Society open science, 5(1), 171545.

[5] Gandhi, S. J., Thakor, M. M., Sheth, J., Pandit, H. I., & Patel, H. S. (2017). Comparison of String Similarity Algorithms to Measure Lexical Similarity. National Journal of System and Information Technology, 10(2), 139.

[6] Hu, Y., Boyd-Graber, J., Satinoff, B., & Smith, A. (2014). Interactive topic modeling. Machine learning, 95(3), 423-469.

[7] Huang, A. (2008, April). Similarity measures for text document clustering. In Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand (pp. 49-56).

[8] Huang, L., Milne, D., Frank, E., & Witten, I. H. (2012). Learning a conceptbased document similarity measure. Journal of the American Society for Information Science and Technology, 63(8), 1593-1608.

[9] Mcallister, R. A., & Angryk, R. A. (2012). Abstracting for Dimensionality Reduction in Text Classification. International Journal of Intelligent Systems,28(2), 115-138. doi:10.1002/int.21543

[10] Patil, L. H., Mrs., & Atique, M., Dr. (2013). A Semantic approach for effective document clustering using WordNet.

[11] Sato, Y., Kawashima, H., Okuda, H., & Oku, M. (2008). Trend-based Document Clustering for Sensitive and Stable Topic Detection.

[12] Sherkat, E., Nourashrafeddin, S., Milios, E. E., & Minghim, R. (2018, March). Interactive Document Clustering Revisited: A Visual Analytics Approach. In 23rd International Conference on Intelligent User Interfaces (pp. 281-292). ACM.

[13] Van der Maaten, L., & Hinton, G. (2008). Visualizing Data using t-SNE. Journal of Machine Learning Research,9, 2579-2605.

[14] Wang, S., & Koopman, R. (2017). Clustering articles based on semantic similarity. Scientometrics, 111(2), 1017-1031.

[15] Xu, W., Liu, X., & Gong, Y. (2003). Document Clustering Based On Non-negative Matrix Factorization.

[16] Scikit-learn developers. Scikit-learn 0.20.0 documentation.

[17] Bhatia, S., Lau, J. H., & Baldwin, T. (2017). An automatic approach for document-level topic model evaluation. arXiv preprint arXiv:1706.05140.

[18] Lang, K. (1995). Newsweeder: Learning to filter netnews. In Machine Learning Proceedings 1995 (pp. 331-339).