

Genre Classification: A Topological Data Analysis Approach

Kevin Shin
Macalester College MATH 471
May 14, 2019

Abstract

Topological Data Analysis (TDA) is a field of applied mathematics in which tools from topology are used to analyze a dataset. The intuition is that by forming a geometric representation that models the data space, important structural features are revealed by analyzing its homological features, which may then have significance in the context of the collected data. Thus, TDA provides an avenue to explore large, potentially high-dimensional data sets in a mathematically rigorous way.

This paper focuses on an application of TDA to Natural Language Processing (NLP), a multidisciplinary area of study which broadly aims to develop computational tools and models to analyze language. By employing NLP techniques to transform text into vectors, I develop a way of analyzing a data set of books downloaded from Project Gutenberg and differentiate them by their listed genre. An algorithm (Zhu 2013) is explored and implemented to capture the “shape of text”, and topological tools are applied to extract the homological features of each book. I discuss two different perspectives on classification (1) a t-test on homological statistics (dimensions of H_1 , feature birth location, etc.) and (2) a k -medoids approach in which clusters of the vector space are formed by associating points close to a medoid. The results of this experiment confirm that TDA provides a powerful way of analyzing text, and I conclude by reflecting on future work necessary to improve the usage of TDA in NLP.

Introduction

We have the intuition that text is structured, and different genres of text or styles of writing yield various structures. Science fiction storylines may involve intricate loops, while adventure narratives may gradually build up to one big climactic event. Classifying and quantifying differences in text is an example of an application of document representation, an area of natural language processing which embodies a group of techniques to represent text data in a way that facilitates computational analysis. The broad goal of this domain is to model natural (human) language in a way that computers can quickly parse and categorize. For example, current natural language processing techniques of document representation include algorithms which take a given text and transforms each unit into a vector, whereby each dimension represents a unique word in the set (Singh et al. 2017).

Topological data analysis (TDA) describes an approach that uses algebraic persistent homology to uncover the underlying structure of data sets. Given a point-set cloud, a sequence of nested simplicial complices are formed by increasing a scale parameter ϵ and recording the “birth” and “death” of homological features which arise and disappear in the sequence. Persistent homology provides a

way of analyzing the “important” structural features by finding the k -dimensional holes (connected components, holes, trapped volumes, etc.) which persist through the scale parameter.

It then seems promising that TDA would be a useful tool in processing text vector spaces, if our intuition is that an important aspect of distinguishing classes of texts is exactly this structural feature. If NLP techniques can accurately model texts into vector spaces, we can treat each vector as a point, forming a point cloud which can then be analyzed by persistent homology.

The goal of this project is to bridge these two domains and provide an example of such an analysis, making the implementation of the computational techniques transparent and easily accessible. To start, this paper will present a short introduction to simplicial and persistent homology, as well as a tour through relevant techniques and ideas in document representation. The following section will describe my experiment, where I apply the NLP and TDA tools introduced to analyze book data. My broad intention is to provide insight into the role TDA can play in language analysis, specifically assessing the success of these methods to distinguish books by their genre.

1 Simplicial Complex Background

A k -simplex is the smallest convex set containing $k + 1$ points in general position, meaning that the vectors formed by the difference of these points are linearly independent in \mathbb{R}^n . For each $k + 1$ -simplex $[v_0v_1v_2\dots v_k]$, a subset of these vertices then forms a sub-simplex of the original simplex. For the $k + 1$ -simplex, we call the k -combinations of its vertices (the k -subsimplicies) a face, and the union of all possible faces forms its boundary. This definition sketches out the first four simplicies:

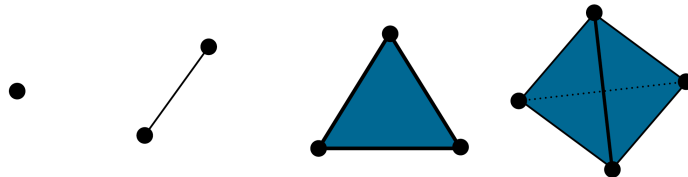


Figure 1: From left to right, the 0-simplex (point), 1-simplex (line segment), 2-simplex (triangle), 3-simplex (tetrahedron).

A simplicial complex $K \subset \mathbb{R}^n$ is defined as a collection of simplicies with the following properties:

- $\sigma \in K, \tau \in \sigma \Rightarrow \tau \in K$
- $\sigma_1, \sigma_2 \in K \Rightarrow \sigma_1 \cap \sigma_2 = \emptyset$ or $\sigma_1 \cap \sigma_2 \subset \sigma_1$ and $\sigma_1 \cap \sigma_2 \subset \sigma_2$

Informally then, a simplicial complex is a combinations of k -simplicies in which all faces of the simplicies are included. For an indepth of simplicial complicies, see (Crossley 2010).

Homology

Homology is a subfield of topology which embodies algebraic methods of distinguishing spaces by counting the number of holes in that space. For instance, we recognize that the circle and disk are not the same shape because the circle has a hole while the disk does not. The intuition is that the

n -dimensional holes offer insight into the structural features of a topological space X —in fact, these holes generate a homology group $H_k(X)$ for a dimension k .

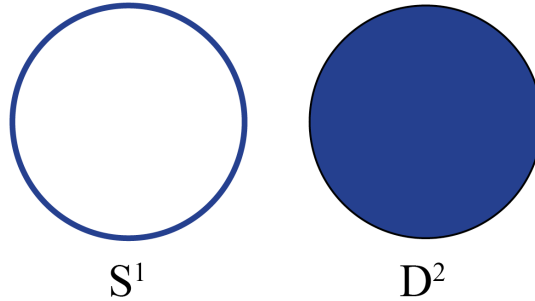


Figure 2: Visual representation of S^1 , the circle, and D^2 , the filled in disc.

Recall that a simplicial complex K is made up of n -simplices. Let $S_n(K)$ denote the set of all n -simplices in K , and $C_n(K)$ be the \mathbb{Z}_2 -vector space with basis $S_n(K)$. Then, we define the boundary operator as :

$$\delta_n : C_n(K) \rightarrow C_{n-1}(K)$$

$$\delta([v_0 v_1 \dots v_n]) \rightarrow \sum_{j=0}^n [v_0 v_1 \dots \hat{v}_j \dots v_n]$$

where \hat{v}_j denotes that v_j is removed from the simplex. δ_n is then a linear transformation on the basis elements of the vector space. The boundary operator can be iteratively applied to any vector space, and we define the **chain complex** as the sequence of chain maps:

$$C_* : \dots C_n \xrightarrow{\delta_n} C_{n-1} \xrightarrow{\delta_{n-1}} C_{n-2} \dots \xrightarrow{\delta_2} C_1 \xrightarrow{\delta_1} C_0 \xrightarrow{\delta_0} 0$$

We define $B_n := \text{Im}(\delta_{n+1}) = \delta_{n+1}(C_{n+1})$ and $Z_n := \text{Ker}(\delta_n)$. Intuitively, Z_n represents cycles and B_n represents boundaries. For a given C_k , B_k and Z_k are subspaces illustrated by the following boundary relationships:

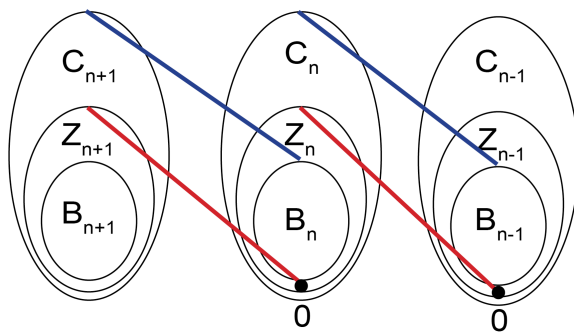


Figure 3: A visual depiction of the n , $n+1$, and $n-1$ vector spaces for some C_n . The blue lines indicate the Image mapping of the boundary operator, that is, $B_n := Im(\delta_{n+1})$. The red lines indicate the Kernel of the δ_n map, or $Z_n := Ker(\delta_n)$. Note also that $B_n \subset Z_n$.

Finally, define $H_n(K) := Z_n(K)/B_n(K)$, the quotient group in which the boundaries are associated together. Thus, $H_n(K)$ represents the n -dimensional cycles that are not boundaries, that is, “holes” in the space.

Persistent Homology

Persistent homology applies the features of simplicial homology on point-clouds by generating a simplicial complex based on the distances between points in the set. The most common methods for building the complex are the Cech complex, the abstract simplicial complex in which unordered $(k+1)$ -tuples form a k -simplex if the $\epsilon/2$ -neighborhoods have a common intersection, and the Vietoris-Rips complex, the abstract simplicial complex in which the points are pairwise within ϵ (Ghrist 2008).

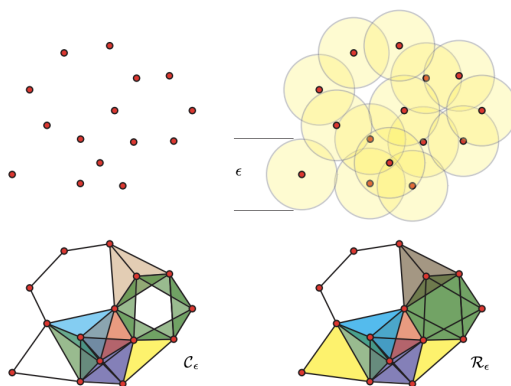


Figure 4: A visual depiction of the Cech and Rips complex (Ghrist 2008).

Note that this definition implies that given a point-cloud, different ϵ -parameters might yield different simplicial complices, and certain ϵ choices will capture different homological features. The insight of persistent homology is to consider all simplicial complices built from a scaling ϵ -parameter, and analyze the homological features which endure for an interval (also known as a lifespan). To this end, the persistent homology of a simplicial complex is often depicted as a persistence diagram, in which a given k -dimensional feature is plotted on an axis that describes its birth time and death time.

2 Natural Language Processing

Natural Language Processing exists as the intersection of various domains of linguistics, mathematics, computer science, and psychology, with the common goal of developing computational techniques which enable computers to process human language (Joshi 1991). A subset of NLP known as document/information retrieval has gained importance as the amount of text information has recently increased dramatically, creating the need for efficient (fast) algorithms and accurate representations of available text (Milius et al. 2019). One of the most popular approaches is the Vector Space Model, where documents are processed as vectors for computational text analysis, and each component is a term within the set of documents.

Vector Space Models

Here we discuss the two most common forms of vectorization, the bag-of-words model and the tf-idf model. To introduce some relevant terminology we'll use in this section, let a **document** be a single body of text; this can be chosen based on the nature of the text analysis (sentences, paragraphs, etc.) A **term** is a single meaningful unit (words, pairs of words, et.). A **corpus** is a collection of documents.

Bag-of-words (BoW)

The Bag-of-words model assumes that the frequency of a term determines its importance in a given document. The vocabulary represents the number of unique words to the corpus, and each document is translated into a vector, where each entry represents the frequency count of the relevant term (Zhao 2017).

Example 1

Let the corpus be the following four lines:

1. I thought well as well
2. him as another and
3. then I asked him with
4. my eyes to ask ¹

The corpus here is the set of these four lines; each line is a document; each word is a term. The basis (dictionary) of the corpus is defined as the unique words of the set:

$$\{I, thought, well, as, him, another, and, then, with, asked, my, eyes, to, ask\}$$

¹Excerpt pulled from Joyce, James. Ulysses. Project Gutenberg. Retrieved May 14, 2019 from <http://www.gutenberg.org/files/4300/4300-h/4300-h.htm>.

. This defines the dimensionality of the space, giving the following four vectors in \mathbb{R}^{14} :

| | I | thought | well | as | him | another | and | then | with | asked | my | eyes | to | ask |
|----------|---|---------|------|----|-----|---------|-----|------|------|-------|----|------|----|-----|
| 1 | 1 | 1 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

TF-IDF

While BoW models have widely been used for many text-analysis computations, the assumptions of the model may lose essential features of the text that are important in uncovering the structure of the text itself. The BoW model assumes that the importance of a term is directly proportional to the number of times it appears in the document, but this implies that sentences in which common words are not the most important words are misrepresented. Additionally, we can easily see the sparsity of the vectors, dubbed the “curse of dimensionality” because the size of the dictionary will likely significantly outweigh the size of the paragraph. This presents a well-known challenge to the computational efficiency of text-analysis algorithms (Milios et al 2019).

TF-IDF (term frequency-inverse document frequency) is a model in which the frequency assumptions are addressed by weighting the word based on its appearance within the document compared to its frequency in the corpus. The intuition here is that words important to the document do appear more times, but this is offset by calculating whether the word appears in other documents, suggesting that it is less important to the particular document itself (Munteau 2007):

$$tfidf(term) = tf(term) \times idf(term)$$

$$tf(term) = \frac{\# \text{ of times term appears in document}}{\# \text{ of terms in document}}$$

$$idf(term) = \ln\left(\frac{\# \text{ of documents}}{\# \text{ of documents in corpus with term}}\right)$$

Although tf-idf does not address the dimensionality problem, the idea is to incorporate a score for each word to assess their relative significance, thereby representing the document more appropriately.

Text Processing Pipeline

In order to process a generic text source, the following cleaning process for the corpus usually takes place:²

1. Parsing/Tokenization: The text is split up into tokens—most commonly, tokens are words.
2. Pruning/Removal of stop-words: some words are used very often in the English language but contain very little semantic meaning (e.g. “the”, “you”, “and”). Including these words unnecessarily increases the dimensionality of the space, so these words are often removed.

²This pipeline is a generalized summary of the process as provided in (Selivanov 2018) and (Manning 2018).

3. Stemming: We further reduce the dimensionality of the space by reducing words to a common base form. For instance, “attacked” and “attacking” are both counted as “attack”.

After cleaning the text, we apply a model to the set (BoW or tf-idf), resulting in a **Document Term Matrix**. Let m be the number of documents and k be the number of terms. Then, the DTM is a $m \times k$ matrix such that the $[i, j]$ entry represents the weight of the j -th term in the i -th document. An example of this was given as a table in the BoW section under Example 1.

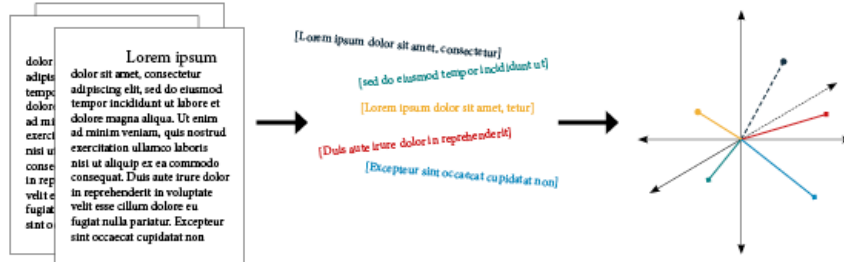


Figure 5: A visual representation of the pipeline of text vectorization,. A set of texts (corpus) is split into documents, which are cleaned and processed. These documents are then vectorized by a chosen technique, where each vector represents a weighted value with respect to a given word present in the set.

Additional Text Analysis Concepts

To summarize, vectorization returns a set of vector representations of paragraphs. In order to apply meaningful analysis to this space, we discuss a notion of distance.

Defining Distance

Persistent homology generates simplicial complicies based on the distances between points in the point cloud. Viewing the space as \mathbb{R}^n perhaps invites us to think of the Euclidean distance between points, but any notion of distance suffices in constructing the complex. Typically in NLP, to capture the notion of similarity, the cosine-similarity distance is used, defined as (Zhu 2013):

$$D(x_i, x_j) = \cos \frac{x_i^T x_j}{\|x_i\| \|x_j\|}$$

From our Document-Term Matrix, we can then compute a cosine distance matrix, where the $[i, j]$ term represents the cosine-distance between the i -th document (i -th row of DTM) and the j -th document (j -th row of DTM). Given k documents, this will be a $k \times k$ matrix, which is symmetric and has diagonals of 0.

Latent Semantic Analysis

The variation in a speaker’s natural vocabulary often leads to high-dimensional vector spaces, even when adjusting for stop-words and stemming. Although these techniques provide an intuitive curtailment of the wordset, there are other, more sophisticated techniques for dimensionality reduction. One of the most common techniques employed for this purpose is Latent Semantic Analysis (LSA), a variation of Principal Component Analysis. The intuition behind the technique is that terms can be related by semantic content not directly present, and so co-occurrences of two terms across documents increase the likelihood that they are related. Mathematically, LSA can be seen as an instance of Singular Value Decomposition applied to document-term matrices.

Let M be a document-term matrix. Then, M can be decomposed into $M = U\Sigma V^T$, such that $U^T U = I$ and $V^T V = I$ and Σ has entries along its diagonal. Then, for a given rank k , M is approximated by A_k by selecting k singular values and reconstructing by multiplying its decomposition. The result of such a decomposition is a semantic space of dimension k , and with a “good” choice of k , the structure of the vector space is preserved. For an in-depth discussion on LSA, see (Gefen et al. 2017).

3 Relevant Works

In “Persistent Homology: An Introduction and a New Text Presentation for Natural Language Processing”, Zhu uses persistent homology to analyze a set of child and adolescent texts, extracting the rank of the H_1 group over the ϵ -range, and the ϵ at which the first H_1 feature appears. Zhu finds statistically significant differences between child and adolescent writing by this simple analysis. Intuitively, one-dimensional holes represent “tie-backs”, perhaps signifying a more mature author capable of relating ideas back to an original thesis or central theme. The key insight of Zhu’s approach is the development of a SIFTS (Similarity Filtration with Time Skeleton) algorithm, a modification of the distance matrix in which documents adjacent to each other in the text are given a distance of 0. When applying persistent homology, the complex starts with a connected skeleton, so $\dim(H_0) = 1$ for all ϵ . Zhu refers to these trivial edges as *time-edges* to capture the flow of a document (Zhu 2013).

In “Topological Signature of 19th Century Novelists: Persistent Homology in Text Mining”, Gholizadeh et al. approach the author-classification problem through topological data analysis methods. They extract the ten most important characters of a given novel using Stanford CoreNLP’s named entity recognizer, and for each character, define an appearance vector in which each entry refers to the index of the character’s appearance in the text. The distance between two characters is defined as the Wasserstein distance of order 0.5 between the two appearance vectors. Each novel then, has 10 character vectors, and the Wasserstein metric of order 1 is applied to analyze the distance between the persistence diagram for each book. This space is used to execute a 5-Nearest Neighbors algorithm in binary classification to predict the author of a book (Gholizadeh et al. 2018).

Reflecting upon these two works, both authors attempt to capture the flow of a document, whether by indexing the appearance of a word or minimizing the distance between adjacent units. Furthermore, Zhu implements what might be seen as a standard vectorization approach in using tf-idf, but the approach of Gholizadeh et al. and the success of their algorithms invites us to consider other, more creative solutions which have the potential of revealing important structural features. Generally, there is little reason to believe that there is a “correct” notion of distance inside these spaces—while Zhu uses cosine-similarity, Gholizadeh et al. use Wasserstein distance.

4 A Topological Approach to Genre Classification

Inspired by works at the intersection of TDA and NLP, this project has the goal of assessing whether persistent homology is successfully able to differentiate between genres of books. NLP techniques mentioned in Section Two are utilized to translate book-texts into vector spaces, and Rips complexes are generated using the cosine-similarity distance. A version of the SIFTS algorithm is directly implemented to modify the resulting distance matrix, and the R package TDA is used to extract homological features of the complex.

4.1 Dataset

The R package `gutenbergr` provides a dataset of public-domain works available for download. An examination of `gutenberg_metadata` provides the following variables:

```
title      author  gutenberg_author_id language  gutenberg_id  language
has_text   gutenberg_bookshelf
```

We first filter the dataset by isolating those files which have full data on title, text, and author. “Children’s Fiction”, “Humor”, “Science Fiction”, “Adventure”, and “Biographies” were chosen as sample genres (`gutenberg_bookshelf`), and a regex matching algorithm was used to sample books from the database, allowing for books with multiple bookshelf categories to be included in the sample.

4.2 Statistical Significance Approach

Each of these genres determines a subset of this dataset. The intuition is that running TDA techniques to return the homological features of these works will give us a sampling distribution (of that feature), and then we can compare these statistics across genres.

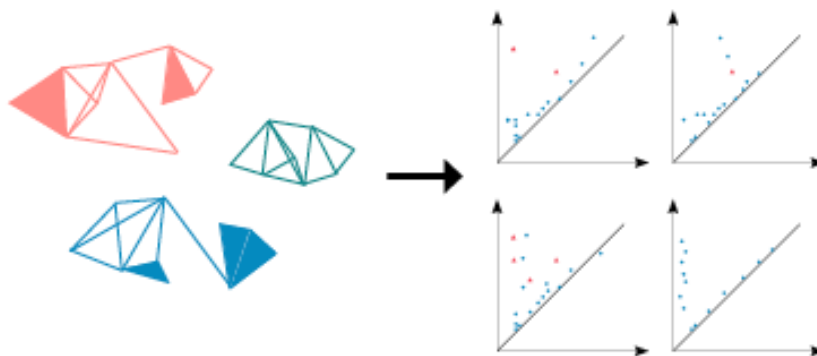
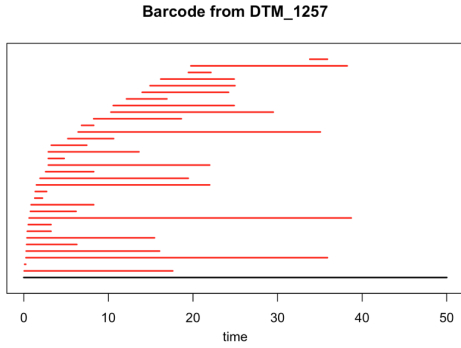


Figure 6: After running persistent homology on our data sets, we will have a set of simplicial complices, each representing one document in our text data. These are then analyzed via persistence diagrams or barcodes to discern their respective homological features.

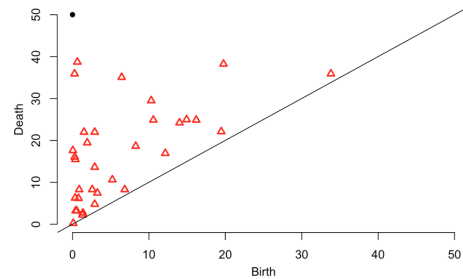
Pipeline, revisited

We sketch the general pipeline to provide an overview of the methods performed on each book. For a more complete treatment or an implementation of the steps, see Appendix B.

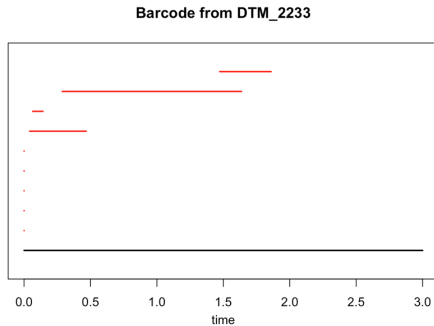
1. Let `bookVector` be a vector of integers representing the `bookID` associated with a book available for download. For this experiment, `bookVector` was created by sampling 45 books from each of the aforementioned categories.
2. For each book, compute its corresponding document-term matrix.
 - Separate the book into paragraphs, and extract the middle $n = 100$.
 - Clean the text of each paragraph.
 - Compute the Document Term Matrix for the book, using tf-idf weighted scores.
 - Perform a LSA to reduce the dimensionality.
3. Compute the distance matrix for the document-term matrix returned from step 2. To reiterate, given a document-term matrix with k paragraphs, the resulting distance matrix is a $k \times k$ -matrix in which the $[i, j]$ term of the matrix represents the cosine-similarity distance between paragraphs i and j . As a result, we expect this matrix to be symmetric and have 0s along its diagonal.
4. To simulate the SIFTS algorithm provided by Zhu (Zhu 2013), this matrix will be modified, where $[i, i + 1]$ and $[i + 1, i]$ terms are set to 0 for all $0 \leq i \leq k - 1$.
5. Run `ripsDiag`, inputting this matrix as the distance matrix. Record homological features reported by the output.



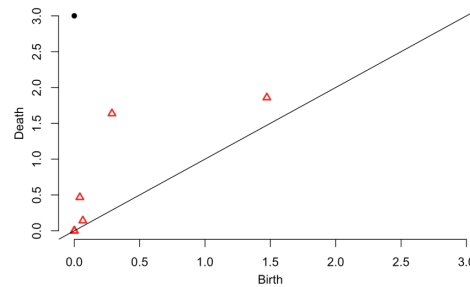
(a) The Barcode for The Three Musketeers



(b) The persistence diagram for The Three Musketeers



(c) The Barcode for A Damsel in Distress



(d) The persistence diagram for A Damsel in Distress

Figure 7: The figures above demonstrate the resulting barcode and persistence diagram completing the pipeline analysis with “The Three Musketeers” (Dumas, `gutenberg_id = 1257`) and “A Damsel in Distress” (PG Wodehouse, `gutenberg_id = 2233`). Note that regardless of the genre, each diagram reports a single connected component (due to the time skeleton), but across the books we see variation in the number of features, the birth time of the feature, and other homological characteristics.

Iterating through this pipeline for each book gives us a dataframe in which one row represents a book, reporting its ID, category, number of features (holes, $\dim = 1$), and average birth location, death location, and lifespan. We consolidate (row-bind) each dataframe to a main data-frame. We compute pairwise t-tests for each genre, for each statistic of interest. To streamline this process, a function was implemented to wrap around the process and output a matrix where an entry represents the p-value returned by the t-test for the genre of the corresponding column and row for a given statistic. See Figure 12, Appendix A and `computeStatistics` in Appendix B.

Analysis

We focus on providing a brief analysis for the average number of H_1 features of a genre. A similar analysis can be repeated for any of the homological statistics included in the data frame. See Figure 13 in Appendix A. Setting $\alpha = 0.05$, there are several pairs of genres which are lower than the significance level for H_1 features:

| Genre 1 | Genre 2 | p-value |
|--------------------|-----------------|----------------|
| Children’s Fiction | Humor | 8.27e-05 |
| Children’s Fiction | Science Fiction | 1.18e-2 |
| Children’s Fiction | Biographies | 1.13e-2 |
| Humor | Adventure | 4.38e-3 |

In terms of a standard statistical analysis of t-tests, it’s important to state the underlying assumptions of the model and interpret the value accurately. The null hypothesis is that the t-test statistic for a given two genres are equivalent. Our p-values denote that for the cases listed above, we are 95% confident that the true difference in the mean of the sample-statistics is not equal to 0. In the context of our experiment, this means that persistent homology on vector spaces generated from tf-idf vectorization and cosine-similarity yield statistically significant values for the number of holes.

Additionally, a second iteration of this pipeline was implemented increasing the number of paragraphs sampled from each book ($n = 200$) (see Figure 14 in Appendix A). We note an increase in the number of significant p-values, and a general decrease in the magnitudes. This may correspond simply to more a greater sample capturing more homological features.

| Genre 1 | Genre 2 | p-value |
|--------------------|-----------------|----------------|
| Children’s Fiction | Humor | 6.44e-09 |
| Children’s Fiction | Science Fiction | 2.29e-03 |
| Children’s Fiction | Adventure | 9.34e-03 |
| Children’s Fiction | Biographies | 6.58e-05 |
| Humor | Science Fiction | 4.69e-03 |
| Humor | Adventure | 1.00e-02 |

Note that Children’s Fiction appears most frequently as a significantly different genre. In light of Zhu’s discussion (Zhu 2013), we might conjecture that topology provides insight into the maturity of the intended author/reader, and that the age of the subject is a significant influence in determining structural or syntactic writing choices. Zhu also posits that H_1 features denote “tie-backs” in the writing—perhaps indicating a structurally advanced style or ability to relate back to aforementioned themes and ideas in the work.

4.3 k -medoids Approach

The statistical significance approach allows us to conclude that there are meaningful differences between some genres, but this statement is limited to the chosen sampling of the dataset. Having completed this initial investigation, we extend our project to include a clustering analysis to assess whether this TDA approach can additionally provide persistent diagrams which can be meaningfully clustered into the genres. Applying a notion of distance between persistence diagrams allows us to run a k -medoids algorithm. Briefly, given a dataset, a medoid is defined as an assignment of one element in the set as a proposed central point. The k -medoid algorithm then assigns k points, minimizing the total dissimilarity distance between points in the set and the medoids, partitioning the set into clusters around these points (Arora 2016). Our goal is to assess whether the clusters correspond to genres, further supporting the claim that structural features captured by persistent homology can meaningfully distinguish between genres.

Analysis

The R Package `cluster` provides the PAM function (Partitioning Around Medoids), where the given input is a distance matrix, and the output is the set of medoids and the clusters around them. We reuse the majority of the pipeline of our previous approach to generate vector space representations of text which are then run through the persistent homology analysis to produce persistence diagrams. Instead of extracting homological features, we instead compute distances between persistent diagrams. For reasons of computational complexity in computing this distance, we shorten the dataset to include 15 books from each genre instead of 45. If we consider k books, the result of this algorithm is a distance matrix of $k \times k$, in which the $[i, j]$ -entry represents the distance between the persistent diagrams of the i -th and j -th book. Similar to the previous section, we expect this to be a symmetric matrix with a diagonal of 0s.

In order to expand our approach to cover different notions of distance, we run the same analysis on four different notions of distance between persistence diagrams: (1) the Bottleneck distance with the Euclidean metric, (2) the Bottleneck distance with the Manhattan metric (3) the Wasserstein distance ($p = 1$) with the Euclidean metric, and (4) the Wasserstein distance ($p = 1$) with the Manhattan metric. Each distance will return a cluster, and we assess the genre of each book, incrementing the tally for each genre. Finally, the data is consolidated into one frame which lists the returned medoids, their category, and the relevant counts, which are normalized.

| medoidID <fctr> | Children's Fiction <dbl> | Humor <dbl> | Science Fiction <dbl> | Adventure <dbl> | Biographies <dbl> | medoidCategory <chr> |
|--------------------|-----------------------------|----------------|--------------------------|--------------------|----------------------|--|
| 5606 | 0.1818182 | 0.2727273 | 0.1363636 | 0.1818182 | 0.2272727 | Adventure |
| 8670 | 0.1739130 | 0.2173913 | 0.2608696 | 0.2173913 | 0.13043478 | Children's Fiction |
| 848 | 0.2666667 | 0.2000000 | 0.2000000 | 0.2666667 | 0.0666667 | Historical Fiction/Children's Literature/Adventure/Movie Books |
| 45532 | 0.4000000 | 0.0000000 | 0.2000000 | 0.0000000 | 0.4000000 | Biographies |
| 3785 | 0.1000000 | 0.1000000 | 0.2000000 | 0.2000000 | 0.4000000 | Children's Fiction |

Figure 8: Bottleneck, Euclidean

| medoidID <fctr> | Children's Fiction <dbl> | Humor <dbl> | Science Fiction <dbl> | Adventure <dbl> | Biographies <dbl> | medoidCategory <chr> |
|--------------------|-----------------------------|----------------|--------------------------|--------------------|----------------------|--|
| 86 | 0.1818182 | 0.2727273 | 0.1363636 | 0.1818182 | 0.2272727 | Precursors of Science Fiction/Arthurian Legends |
| 8670 | 0.1818182 | 0.1818182 | 0.2727273 | 0.2272727 | 0.1363636 | Children's Fiction |
| 530 | 0.3750000 | 0.0000000 | 0.1250000 | 0.1250000 | 0.3750000 | Children's Fiction |
| 848 | 0.2307692 | 0.3076923 | 0.2307692 | 0.2307692 | 0.0000000 | Historical Fiction/Children's Literature/Adventure/Movie Books |
| 3785 | 0.1000000 | 0.1000000 | 0.2000000 | 0.2000000 | 0.4000000 | Children's Fiction |

Figure 9: Bottleneck, Manhattan

| medoidID <fctr> | Children's Fiction <dbl> | Humor <dbl> | Science Fiction <dbl> | Adventure <dbl> | Biographies <dbl> | medoidCategory <chr> |
|--------------------|-----------------------------|----------------|--------------------------|--------------------|----------------------|-------------------------|
| 24313 | 0.1538462 | 0.2307692 | 0.19230769 | 0.1538462 | 0.2692308 | Humor |
| 9446 | 0.2727273 | 0.1818182 | 0.09090909 | 0.2727273 | 0.1818182 | Humor |
| 7251 | 0.5000000 | 0.1666667 | 0.0000000 | 0.1666667 | 0.1666667 | Humor |
| 6903 | 0.2000000 | 0.1333333 | 0.3333333 | 0.2000000 | 0.1333333 | Science Fiction |
| 11870 | 0.1176471 | 0.2352941 | 0.2352941 | 0.2352941 | 0.1764706 | Science Fiction |

Figure 10: Wasserstein, Euclidean

| medoidID <fctr> | Children's Fiction <dbi> | Humor <dbi> | Science Fiction <dbi> | Adventure <dbi> | Biographies <dbi> | medoidCategory <chr> |
|--------------------|-----------------------------|----------------|--------------------------|--------------------|----------------------|---|
| 20533 | 0.1538462 | 0.2307692 | 0.1923077 | 0.1538462 | 0.2692308 | Humor |
| 7862 | 0.5000000 | 0.1250000 | 0.0000000 | 0.2500000 | 0.1250000 | US Civil War/Children's Fiction |
| 9446 | 0.2222222 | 0.2222222 | 0.1111111 | 0.2222222 | 0.2222222 | Humor |
| 6903 | 0.2142857 | 0.1428571 | 0.3571429 | 0.1428571 | 0.1428571 | Science Fiction |
| 42 | 0.1111111 | 0.2222222 | 0.2222222 | 0.2777778 | 0.1666667 | Horror/Movie Books/Precursors of Science Fiction/Gothic Fiction |

Figure 11: Wasserstein, Manhattan

The analysis is inconclusive to suggest that the clusters effectively capture the genre of text. Note that only Children’s Fiction, using the Wasserstein-Manhattan or Wasserstein-Euclidean distance, yields a hit-ratio of 0.5 accuracy ; otherwise, we see serious or ambiguous miscategorization. The variation in the data combined with the low accuracy rates suggest that the distance between persistence diagrams may not carry meaningful structural implications for text genres. This lack of significance may also be explained by the limiting features of the dataset and computational complexity conducted within the analysis—using 15 books hardly counts as a “representative” sample of each genre, compared to the number of books available.

5 Conclusions

Topological data analysis provides a powerful way of analyzing Vector Space Models of text from document representation techniques in natural language processing. The results of assessing the statistical significance of various homological features provides a framework of understanding genre through structural features of text-documents. In our work of extending this topological approach to a k -medoids clustering, we failed to see a significantly recognizable cluster of books of particular genres.

In light of these results, further work is needed to explore the classification power of persistent diagrams. This area could expand by minimizing computational complexity, running a similar analysis on a larger, more comprehensive dataset of texts. For a comparison of current persistence representations, see (Adams 2017). We might also employ other document representation techniques which may be better suited to capture genre, or switch dimensionality reduction techniques. This project aims to provide the general framework and establish that topological data analysis and natural language processing intersect in a meaningful and promising area of research.


Acknowledgements: This paper was submitted (5/14/19) as a project for Topology (MATH 471), taught by Lori Ziegelmeier at Macalester College. I would like to thank her for her constant support, guidance, and instruction. Any errors should be attributed only to the original author. Unless otherwise explicitly stated and cited, the artwork appearing in this work is original.

References

- [1] Adams, H., Emerson, T., Kirby, M., Neville, R., Peterson, C., Shipman, P., et al. (2017). Persistence Images: A Stable Vector Representation of Persistent Homology. *Journal of Machine Learning Research*, 18, 135.
- [2] Arora, Preeti , Deepali Varshney, Shipra. (2016). Analysis of K-Means and K-Medoids Algorithm For Big Data. *Procedia Computer Science*. 78. 507-512. 10.1016/j.procs.2016.02.095.
- [3] Crossley, M. D. (2010). *Essential topology*. London: Springer.
- [4] Feinerer, I. (2018, December 21). Introduction to the tm Package Text Mining in R. <https://cran.r-project.org/web/packages/tm/vignettes/tm.pdf>. Accessed 14 May 2019
- [5] Gefen, D., Endicott, J. E., Fresneda, J. E., Miller, J., Larsen, K. R. (2017). A Guide to Text Analysis with Latent Semantic Analysis in R with Annotated Code: Studying Online Reviews and the Stack Exchange Community. *Communications of the Association for Information Systems*, 41, pp-pp. <https://doi.org/10.17705/1CAIS.04121>
- [6] Gholizadeh, S., Seyeditabari, A., Zadrozny , W. (2018). Topological Signature of 19th Century Novelists: Persistent Homology in Text Mining. *Big Data and Cognitive Computing*, 2(4). doi:<https://doi.org/10.3390/bdcc2040033>
- [7] Ghrist, R. (2008). BARCODES: THE PERSISTENT TOPOLOGY OF DATA. *Bulletin of the American Mathematical Society*, 45, 6175. doi:<https://doi.org/10.1090/S0273-0979-07-01191-3>
- [8] Harish, B S Guru, Devanur Shantharamu, Manjunath. (2010). Representation and Classification of Text Documents: A Brief Review. *International Journal of Computer Applications, Special Issue on RTIPPR*. 1. 110 - 119.
- [9] Joshi, A. (1991). Natural Language Processing. *Science*, 253(5025), 1242-1249. Retrieved from <http://www.jstor.org/stable/2879169>
- [10] Manning, C. D., Raghavan, P., Schtze, H. (2018). *Introduction to information retrieval*. Cambridge: Cambridge University Press.
- [11] Milios, Evangelos Shafiei, Mahdi Wang, Singer Zhang, Roger. (2019). A Systematic Study on Document Representation and Dimensionality Reduction for Text Clustering.
- [12] Munteau, Dan. (2007). Vector space model for document representation in information retrieval. *Annals of Dunarea de Jos*. 2007.
- [13] Selivanov, D. (2018, January 11). Analyzing Texts with the text2vec package. <https://cran.r-project.org/web/packages/text2vec/vignettes/text-vectorization.html>. Accessed 14 May 2019
- [14] Silge, J., Robinson, D. (2019, March 23). *Text Mining with R*. [Text Mining with R](https://www.tidytextmining.com/index.html). <https://www.tidytextmining.com/index.html>. Accessed 14 May 2019
- [15] Singh, K. N., Mahanta, A. K., Devi, H. H. M. (2017). Document representation techniques and their effect on the document Clustering and Classification: A Review. *International Journal of Advanced Research in Computer Science; Udaipur*, 8(5). Accessed 13 May 2019

- [16] Zhao, Rui Mao, Kezhi. (2017). Fuzzy Bag-of-Words Model for Document Representation. IEEE Transactions on Fuzzy Systems. PP. 1-1. 10.1109/TFUZZ.2017.2690222.
- [17] Zhu, Xiaojin. (2013). Persistent homology: An introduction and a new text representation for natural language processing. IJCAI International Joint Conference on Artificial Intelligence. 1953-1959.

Appendix A: Figures



| BOOKID | CATEGORY | HOLES | AVG BIRTH | AVG DEATH | AVG LIFE |
|--------|----------|-------|-----------|-----------|----------|
| 472 | Humor | 4 | 0.542464 | 0.877624 | 0.341535 |

| BOOKID | CATEGORY | HOLES | AVG BIRTH | AVG DEATH | AVG LIFE |
|--------|--------------------|-------|-----------|-----------|----------|
| 472 | Humor | 4 | 0.542464 | 0.877624 | 0.341535 |
| 753 | Adventure | 14 | 0.53153 | 0.753642 | 0.231513 |
| 972 | Children's Fiction | 51 | 0.16145 | 0.4513 | 0.3511 |
| 51557 | ... | ... | ... | ... | ... |
| 5142 | Humor | 8 | 0.1561 | 0.64151 | 0.53151 |
| 875 | Biographies | 1 | 0.43143 | 0.5314141 | 0.13415 |

Figure 12: A visual representation of the iterative analysis employed to return the final data set. The blue table represents an instantiation of a single book, with the appropriately extracted homological statistics. This frame is then appended to a global dataframe.

```

> holes_100
Children's Fiction      Humor Science Fiction  Adventure Biographies
Children's Fiction    1.000000e+00  8.268592e-05    0.01184958  0.499237896  0.01133204
Humor                 8.268592e-05  1.000000e+00    0.14792516  0.004378056  0.57552301
Science Fiction       1.184958e-02  1.479252e-01    1.00000000  0.114516667  0.56410400
Adventure             4.992379e-01  4.378056e-03    0.11451667  1.000000000  0.06821425
Biographies           1.133204e-02  5.755230e-01    0.56410400  0.068214255  1.00000000

> life_100
Children's Fiction      Humor Science Fiction  Adventure Biographies
Children's Fiction    1.000000000  0.001538877    0.05639127  0.54679972  0.1957286
Humor                 0.001538877  1.000000000    0.20646966  0.04155657  0.3795358
Science Fiction       0.056391269  0.206469658    1.00000000  0.32097913  0.9865884
Adventure             0.546799723  0.041556570    0.32097913  1.00000000  0.4651158
Biographies           0.195728646  0.379535767    0.98658845  0.46511584  1.0000000

> birth_100
Children's Fiction      Humor Science Fiction  Adventure Biographies
Children's Fiction    1.000000000  0.001105291    0.08746382  0.66602234  0.29816366
Humor                 0.001105291  1.000000000    0.13667312  0.01004607  0.08448749
Science Fiction       0.087463824  0.136673123    1.00000000  0.24778600  0.66666908
Adventure             0.666022338  0.010046069    0.24778600  1.00000000  0.54300004
Biographies           0.298163657  0.084487493    0.66666908  0.54300004  1.0000000

> death_100
Children's Fiction      Humor Science Fiction  Adventure Biographies
Children's Fiction    1.000000000  0.03609197    0.3337798  0.90152452  0.7370705
Humor                 0.03609197  1.000000000    0.2652800  0.07834344  0.1300216
Science Fiction       0.33377984  0.26527995    1.0000000  0.45426878  0.6011951
Adventure             0.90152452  0.07834344    0.4542688  1.0000000  0.8438613
Biographies           0.73707049  0.13002157    0.6011951  0.84386125  1.0000000

```

```

> holes_200
Children's Fiction      Humor Science Fiction  Adventure Biographies
Children's Fiction    1.000000e+00 6.441391e-09    0.002292265 0.009341154 6.583919e-05
Humor                 6.441391e-09 1.000000e+00    0.004690865 0.010033933 5.429677e-01
Science Fiction       2.292265e-03 4.690865e-03    1.000000000 0.939259923 9.368631e-02
Adventure             9.341154e-03 1.003393e-02    0.939259923 1.000000000 1.073971e-01
Biographies           6.583919e-05 5.429677e-01    0.093686313 0.107397088 1.000000e+00
> life_200
Children's Fiction      Humor Science Fiction  Adventure Biographies
Children's Fiction    1.000000e+00 2.456459e-05    0.05802044 0.01704443 0.0000919213
Humor                 2.456459e-05 1.000000e+00    0.03624945 0.13339565 0.5653229768
Science Fiction       5.802044e-02 3.624945e-02    1.000000000 0.60428033 0.0224790491
Adventure             1.704443e-02 1.333957e-01    0.60428033 1.000000000 0.0723938292
Biographies           9.192130e-05 5.653230e-01    0.02247905 0.07239383 1.0000000000
> birth_200
Children's Fiction      Humor Science Fiction  Adventure Biographies
Children's Fiction    1.000000e+00 2.693659e-06    0.01070650 0.006330619 0.001734608
Humor                 2.693659e-06 1.000000e+00    0.03375409 0.147603040 0.512334531
Science Fiction       1.070650e-02 3.375409e-02    1.000000000 0.634612907 0.262863032
Adventure             6.330619e-03 1.476030e-01    0.63461291 1.000000000 0.524185940
Biographies           1.734608e-03 5.123345e-01    0.26286303 0.524185940 1.000000000
> death_200
Children's Fiction      Humor Science Fiction  Adventure Biographies
Children's Fiction    1.0000000000 0.0001098258    0.02112508 0.01555295 0.05052051
Humor                 0.0001098258 1.0000000000    0.09792822 0.24209520 0.20722600
Science Fiction       0.0211250754 0.0979282216    1.000000000 0.72032803 0.90337877
Adventure             0.0155529478 0.2420952003    0.72032803 1.000000000 0.84991704
Biographies           0.0505205136 0.2072260014    0.90337877 0.84991704 1.00000000

```

Figure 14: The same computation done for Figure 9, with $n = 200$ paragraphs.

Appendix B: Selections of code³

```
###Create DTM
computedTM <- function(IDNum, lineLimit){
  book <- gutenberG_download(IDNum, strip=TRUE)
  book <- transmute(book, gutenberG_id = gutenberG_id,
  text = iconv(book$text,"UTF-8","UTF-8",sub=''))
  book <- unnest_tokens(book,input="text",output="Paragraph",token="paragraphs")
  book <- book[50:(50+lineLimit),]
  book <- book %>% filter(!is.na(book$Paragraph))
  book <- VCorpus(VectorSource(book$Paragraph))
  book <- tm_map(book, stripWhitespace)
  book <- tm_map(book, content_transformer(tolower))
  book <- tm_map(book, removeWords, stopwords("english"))
  book <- tm_map(book, removePunctuation)
  book <- tm_map(book, stemDocument, language = "english")
  dtm <- DocumentTermMatrix(book)
  dtm <- weightTfIdf(dtm)
  if ((dim(dtm)[2]) != 0){
    dtmLSA <- lsa(dtm, dims=dimcalc_share())$tk
    return(dtmLSA)
  } else {
    return(NULL)
  }
}

###makeDistanceMatrix
makeDistanceMatrix <- function(datamatrix){
  distanceMatrix <- datamatrix
  cosineDistMatrix <- cosSparse(distanceMatrix)
  diag(cosineDistMatrix) <- 0
  for(row in 2:nrow(cosineDistMatrix)) {
    #implementation of Zhu SIFTS algorithm
    cosineDistMatrix[row, row-1] <- 0
    cosineDistMatrix[row-1, row] <- 0
  }
  for(row in 1:nrow(cosineDistMatrix)) {
    for(col in 1:nrow(cosineDistMatrix)) {
      #scale to see features clearly
      cosineDistMatrix[row,col] = abs(cosineDistMatrix[row,col])*1000000000000000000
    }
  }
  return(as.matrix(cosineDistMatrix))
}
```

³In the interest of providing clear and transparent implementation of code, functions have been selected specifically for reference from portions of the paper. For the interested reader, the full repository is available at <https://github.com/kevin-shin/TopologyNLP>.

```

}

###Main function
mainDataFrame <- data.frame(matrix(ncol = 8, nrow = 0))
columns <- c("bookID", "numHoles", "category", "averageBirthLocation",
"birthLocationSD", "averageDeathLocation", "deathLocationSD", "avgLengthofLife")
colnames(mainDataFrame) <- columns
#generate the main dataframe
main <- function(numLines){
  for (category in topics){
    bookShieldIDs <- isolateIDs(category)
    for (value in bookShieldIDs){
      bookDTM <- computedDTM(value,numLines)
      print(value)
      if (!is.null(bookDTM)){
        bookDistanceMatrix <- makeDistanceMatrix(bookDTM)
        Diag <- ripsDiag(X = bookDistanceMatrix,
          maxdimension = 1,
          maxscale = 100,
          dist = "arbitrary",
          library = "Dionysus",
          printProgress = FALSE,
          location=TRUE)

        numHoles <- summary.diagram(Diag[["diagram"]])$n
        print(numHoles)
        tempDF <- data.frame(value, numHoles, category, mean(Diag$birthLocation),
          sd(Diag$birthLocation), mean(Diag$deathLocation), sd(Diag$deathLocation),
          mean(Diag$deathLocation- Diag$birthLocation))
        names(tempDF) <- c("bookID", "numHoles", "category", "averageBirthLocation",
          "birthLocationSD", "averageDeathLocation", "deathLocationSD", "avgLengthofLife")
        mainDataFrame <- rbind(mainDataFrame,tempDF)
      }
      else {
        print("0 here")
        tempDF <- data.frame(value, 0, category, 0, 0, 0, 0, 0)
        names(tempDF) <- c("bookID", "numHoles", "category", "averageBirthLocation",
          "birthLocationSD", "averageDeathLocation", "deathLocationSD", "avgLengthofLife")
        mainDataFrame <- rbind(mainDataFrame,tempDF)
      }
    }
  }
}
return(mainDataFrame)
}

###return matrix of p-values. Statistic of interest must be replaced inside the function
compareStatistics <- function(dataFrame){

```

```

df <- data.frame(matrix(0, ncol = 5, nrow = 5))
colnames(df) <- topics
rownames(df) <- topics
for (row in rownames(df)){
  for (col in colnames(df)){
    genreRowTable <- dataFrame %>% filter(category == row)
    genreColTable <- dataFrame %>% filter(category == col)
    significanceLevel <- t.test(genreRowTable$averageDeathLocation,
    genreColTable$averageDeathLocation)$p.value
    df[row,col] <- significanceLevel
  }
}
return(df)
}

### EXAMPLE: return a distance matrix for the persistence diagrams
### Wasserstein, Manhattan
collection <- bookList
clusteringPD <- function(numLines){
  similarityMatrix <- matrix(ncol = length(collection), nrow = length(collection))
  diag(similarityMatrix) <- 0
  for (bookIndexHoriz in 1:length(collection)){
    for (bookIndexVert in 1:length(collection)){
      if (bookIndexVert > bookIndexHoriz){
        bookDTM <- computeDTM(collection[bookIndexVert],numLines)
        otherBookDTM <- computeDTM(collection[bookIndexHoriz],numLines)
        if (!is.null(bookDTM) && !is.na(otherBookDTM)){
          bookDistanceMatrix <- makeDistanceMatrix(bookDTM)
          otherBookDistanceMatrix <- makeDistanceMatrix(otherBookDTM)
          Diag <- ripsDiag(X = bookDistanceMatrix,
            maxdimension = 1,
            maxscale = 100,
            dist = "arbitrary",
            library = "Dionysus",
            printProgress = FALSE,
            location=TRUE)
          DiagOther <- ripsDiag(X = otherBookDistanceMatrix,
            maxdimension = 1,
            maxscale = 100,
            dist = "arbitrary",
            library = "Dionysus",
            printProgress = FALSE,
            location=TRUE)
          wassersteinDist <- wasserstein(Diag[["diagram"]], DiagOther[["diagram"]],
            p = 1, dimension = 1)
          similarityMatrix[bookIndexHoriz,bookIndexVert] <- wassersteinDist
        }
      }
    }
  }
}

```

```

    }
  }
  else {
    similarityMatrix[bookIndexHoriz,bookIndexVert] <- 0
  }
}
}
mainMatrix <- similarityMatrix + t(similarityMatrix)
toReturn <- as.data.frame(mainMatrix)
colnames(toReturn) <- collection
rownames(toReturn) <- collection
return(toReturn)
}

similarityTable <- clusteringPD(100)
pamx <- pam(similarityTable, 5, diss = FALSE, cluster.only = FALSE, metric = "manhattan")

### EXAMPLE: Run clustering algorithm
# BottleNeck, Euclidean

bottleNeckEuclidean_15 <- pamx

evaluateClusters <- function(dataframe,i){
  return(row.names(subset(dataframe,widths.cluster == i)))
}

clusteringScore <- function(dataframe){
  df <- data.frame(matrix(0L,nrow = length(unique(dataframe$widths.cluster)), ncol = 5))
  colnames(df) <- topics
  rownames(df) <- unique(dataframe$widths.cluster)
  for (i in rownames(df)){
    clusterVector <- evaluateClusters(dataframe,i)
    for (value in clusterVector){
      relBook <- gutenbergs_metadata %>% filter(gutenberg_id == value)
      category <- relBook$gutenberg_bookshelf
      for (genre in colnames(df))
        if (grepl(genre,category)){
          df[i,genre] <- df[i,genre] + 1
        }
    }
  }
}
normalized <- normalize.rows(df)
colnames(normalized) <- topics
rownames(normalized) <- unique(dataframe$widths.cluster)

```

```
    return(as.data.frame(normalized))
  }

returnScores <- function(dataframe){
  clusters <- clusteringScore(as.data.frame(dataframe$silinfo))
  df <- data.frame("medoidID" = row.names(dataframe$medoids))
  df <- cbind(df,clusters)
  df <- df %>% mutate("medoidCategory" = "temp")
  for (i in 1:nrow(df)){
    df$medoidCategory[i] <- getCategory(df$medoidID[i])
  }
  return(df)
}
```