TOPOLOGICAL DATA ANALYSIS IN TEXT PROCESSING

by

Shafie Gholizadeh

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Computing & Information Systems

Charlotte

2020

ABSTRACT

SHAFIE GHOLIZADEH. Topological Data Analysis in Text Processing. (Under the direction of DR. WLODEK ZADROZNY)

Topological Data Analysis denotes the set of algorithms and methods to define and retrieve the underlying structure of the shapes in the data. Utilizing topological inference in data mining and generally data science is recent, while computational geometry and computational topology have been examined in the area of applied mathematics for many years. Some recent studies have shown the strength of topological data analysis when dealing with high-dimensional data sets. Dealing with the noisy data, the most common goal in TDA is to refine the underlying shapes as the most important property of the data. Then what remains may be considered irrelevant information or simply the noise. Topological inference has been applied to many sub-areas of pattern recognition and data mining, but it is not widely used in natural language processing and text mining. A simple reason is that defining shapes in the text is not easy. In this dissertation document, we introduce three different algorithms of extracting topological features from textual documents, using as the underlying representations of text the two most popular methods, namely term frequency vectors and word embeddings, and also without using any conventional features: (1) To extract topological features without using conventional features, we analyze the graph of appearance/co-appearance of different entities through long documents. We show how these topological structures in a text may effectively act as the signature or identifier of the topic, writer, writing, etc. (2) Then we introduce a new algorithm of extracting topological features from text, namely by converting a sequence of word embeddings into a time series, and analyzing the dimensions of the resulting series for topological persistence. (3) We also provide a topological method to analyze the geometry of the term frequency space. In all three algorithms, we apply homological persistence to reveal the geometric structures under different distance resolutions. We focus on utilizing our defined features for text classification, though they may be useful for other natural language processing tasks as well. Our results show that even if the representation of documents is derived from the standard term frequency matrix or word embeddings space, similarly produced topological features improve the accuracy of classification, meaning that our topological features carry some exclusive information that is not captured by conventional text analysis methods.

ACKNOWLEDGEMENTS

I would like to thank my PhD advisor Dr. Wlodek Zadrozny, and my dissertation committee members, Dr. Jing Yang, Dr. Samira Shaikh, Dr. Erik Saule, and Dr. Rajib Paul. I would also like to express my gratitude to the graduate school of the University of North Carolina at Charlotte, the Department of Computer Science, School of Data Science, Belk College of Business, and the Department of Mathematics and Statistics, for their support and for all I have learnt during my PhD study.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

BOW An acronym for Bag Of Words.

CNN An acronym for Convolutional Neural Network.

GloVe An acronym for Global Vectors.

kNN An acronym for k Nearest Neighbors

LSTM An acronym for Long Short Term Memory.

NER An acronym for Name Entity Recognizer.

RNN An acronym for Recurrent Neural Network.

TDA An acronym for Topological Data Analysis.

CHAPTER 1: INTRODUCTION

The most common goal in TDA is to refine the underlying shapes as the main property of the data. Recent TDA studies have often targeted pattern recognition, dimensionality reduction, or clustering. As high dimensional large data sets are usually represented data clouds, data mining methods especially distance-based algorithms try to study the data cloud and reveal, analyze and summarize the relations among the points in the cloud. Maybe the simplest way to show that how a large number of data points can represent the whole space is to construct the Voronoi Diagram [1]. Voronoi diagram partitions the space to convex sub-regions (e.g., polygons, polyhedrons, etc) and each sub-region covers the area in which a site (i.e., a particular data point) is the closest data point. An example of the Voronoi diagram is illustrated in Figure 1.2. TDA methods usually introduce a similar intuition but follow more complex methodologies. While the geometry (as offered by Voronoi Diagrams) depends on distances to study shapes, topology benefits from homeomorphisms that are closed with respect to stretching or shrinking. Therefore, TDA methods are often much less sensitive to the change of metrics [2]. This will allow us to use TDA for a wide family of problems in which the shape of data matters.

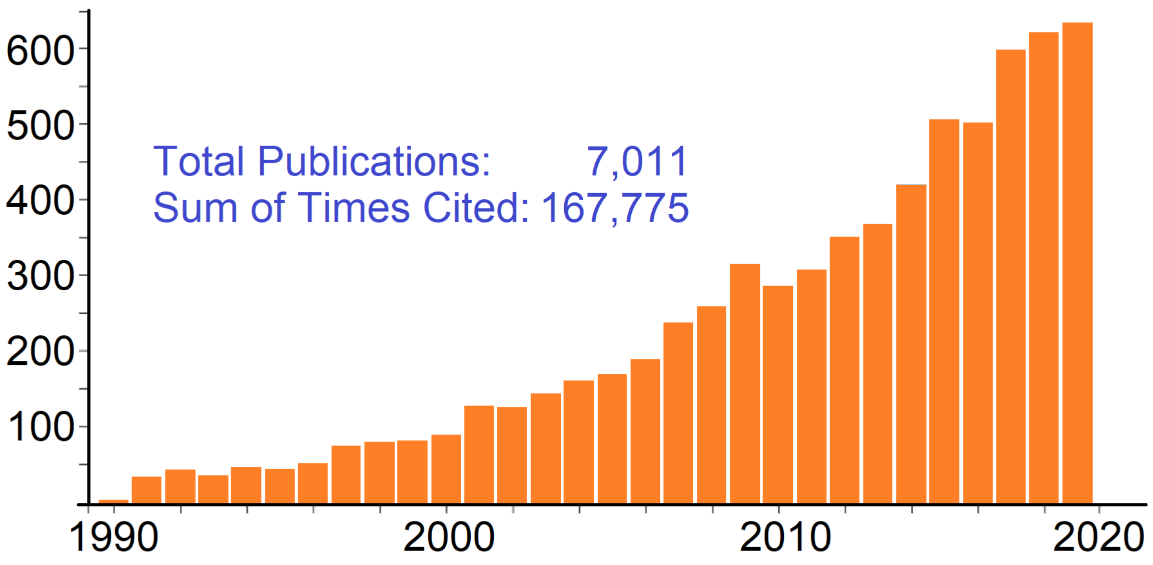
Using topological inference in data mining and generally, data science is recent, while computational geometry and computational topology have been examined in the area of applied mathematics for many years. Figure 1.1 illustrates TDA number of academic publications per year and the publication topics[1]. In some recent studies,

[1]Web of Science, retrieved on April 20, 2020

the topological inference has been used as an alternative to conventional data analysis and artificial intelligence methods. Particularly, TDA has been examined to deal with high-dimensional data sets and/or noisy data sets. Dealing with the noisy data, the most common goal in TDA is to refine the underlying structure as the most important property of the data. Then what remains may be considered irrelevant information or simply the noise.

Large data sets often contain a huge number of discrete points. Therefore, we may require a technique to transform them to some meaningful and continuous topological representation, intuitively that would be like the visual interpretation what may distinguish a continuous meaningful shape out of some close yet discrete points (e.g., connecting a few stars and interpreting them as the Great Bear). In topological data analysis, the tool that provides this kind of interpretation is referred to as *persistent homology* [3, 4, 5, 6]. Here, homology refers to the loops or holes in a topological structure at each dimension. The number of holes at different dimensions is usually all we are interested to know about the topological properties of shapes, though these numbers do not represent further geometric properties.

TDA is not widely utilized in natural language processing and text mining methods. A simple reason is that of course defining the shapes in the text is not easy. Here we introduce some algorithms to define some underlying structures in the text and capture their shapes. An initial effort would be to investigate what is missing or not fully covered by conventional text mining approaches and then try to utilize topological inference to address them. For instance, in many text mining methods, the orders appearance and co-appearance of different entities are not considered. If we assume those lost orders may carry useful information for the analysis, TDA can possibly provide some framework to analyze the underlying structure of those orders in the natural language and contribute to the state of the art in natural language



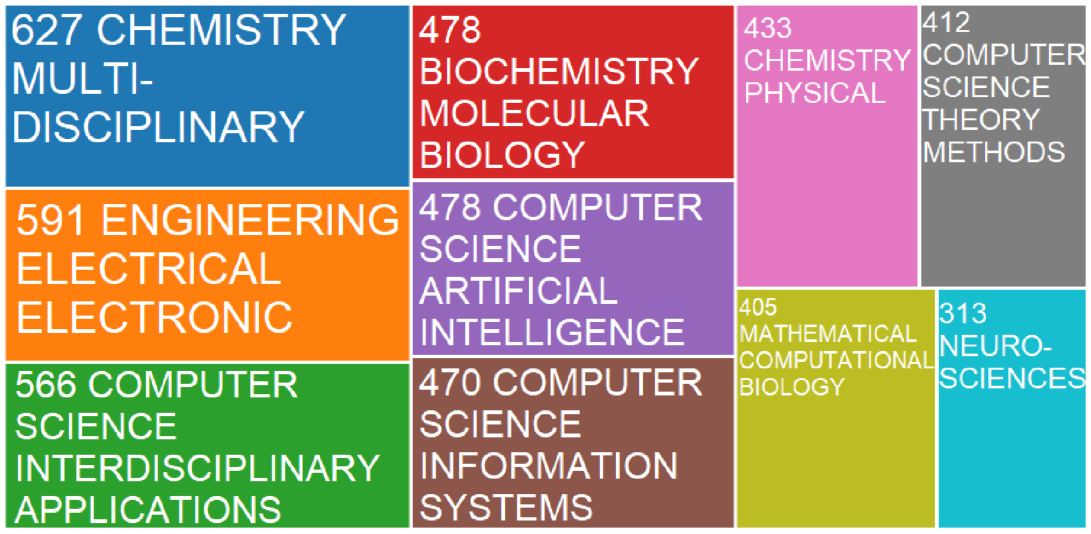


Figure 1.1: Publications on topological data analysis 1990-2019 based on Web of Science data (top) and TDA publication topics (bottom).

processing, by providing some additive information regarding the natural language data. In this dissertation document, we will look at the recent developments and contributions in topological data analysis and then we will introduce and introduce and examine a series of methods in the application of persistent homology in text

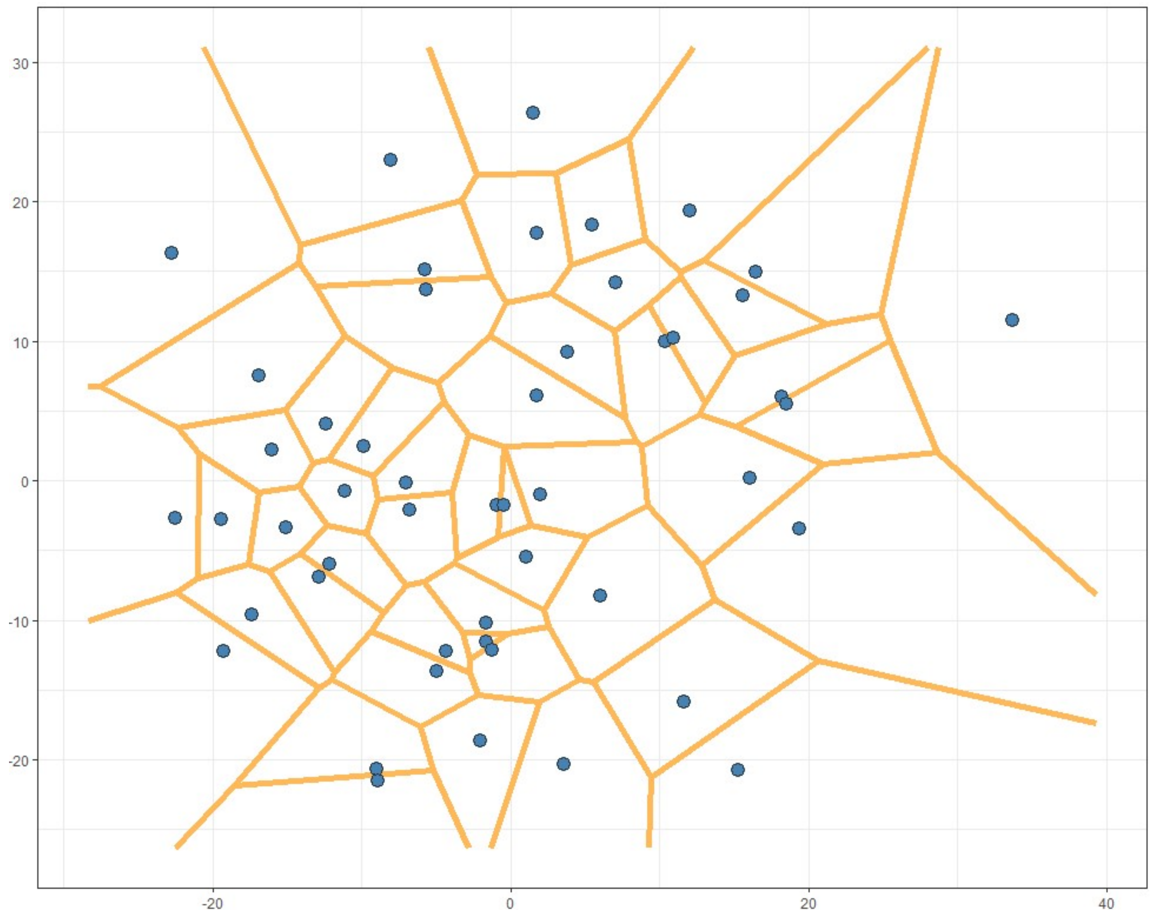


Figure 1.2: An example of Voronoi diagram in 2D Euclidean system.

processing: (a) We will introduce different ways of extracting topological features from textual documents. (b) Then we will examine how to use those features for text classification. (c) We will investigate how the topological structures in a text may effectively act as the signature or identifier of the topic, writer, writing, etc. In other words, (d) we generally show the value of TDA techniques in text processing.

We will introduce some theoretical background in Chapter 2. Chapter 3 describes recent developments in topological data analysis and specially persistent homology. In Chapter 4 we describe our methodology, including three different algorithms to extract topological features (1) without using any conventional text representation, (2) from term frequency space, and (3) from word embeddings representation of text.

Summary of the results and contributions, along with the possible future directions are mentioned in Chapter 5.

CHAPTER 2: BACKGROUND

A topology on a set is defined as the collection of subsets containing the empty set and the whole set that is closed under intersection and union operations [7]. Note that by definition, topology is not generally closed under complementing. This is considered the only difference between topology and σ-algebra. Instead, we call any element in a topology and the complement of any element in topology an *open set* and a *closed set* respectively.

In TDA, the data cloud is usually being studies based on the underlying *simplicial complexes*. We may use the notation of *simplex* for a single data point (referred to as 0-simplex), a connected pair of data records (1-simplex), three pair-wise connected data records (2-simplex), etc. In general, a fully connected set of $(p+1)$ data records is referred to as a p-simplex. For the consistency of our definition we let (-1)-simplex denote an empty set [7]. Now we can define a simplicial complex as a set of some simplices, given that all the subsets of any simplex belonging to the simplicial complex are also present in the same simplicial complex. Some examples of the simplices and an instance of the simplicial complex are illustrated in Figure 2.1.

On a high dimensional space where the dimension of simplices could be high, we usually need to reduce the dimensions. Moving from the highest dimensions and decreasing the dimensions step-wise, we will get a Chain Complex $(C_n : n = p, p-1, \ldots, 1)$ where C_p denotes the set of all p-simplices. Then we can define the *Euler Characteristic* for a data cloud (here the simplicial complexes), as in Equation 2.1 [8, 2].

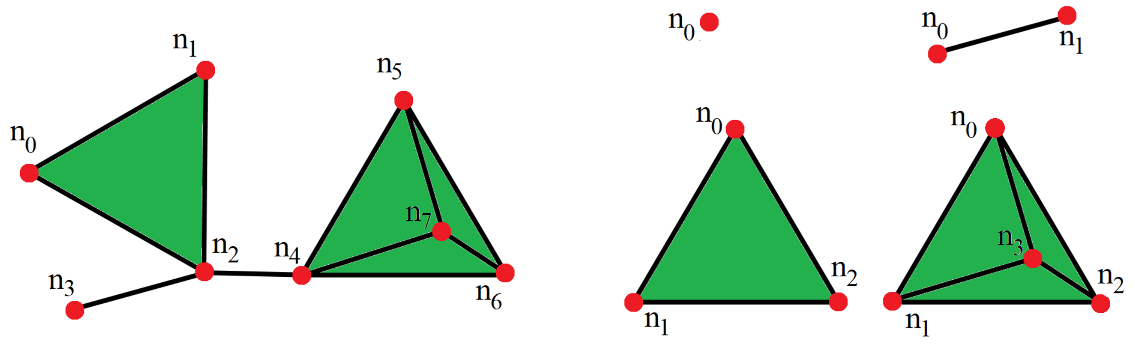


Figure 2.1: An instance of the simplicial complex (left) and some examples of simplices (right).

$$\begin{aligned}\chi(C) &= \sum_p (-1)^p \; dim \; C_p \\ &= \sum_p (-1)^p \text{ number of p-simplices} \\ &= \sum_p (-1)^p \; dim \; H_p\end{aligned} \tag{2.1}$$

We define the p^{th} chain group C_p as the free Abelian group of the oriented p-simplices. Thus, any element c in C_p is a p-chain satisfying $c = \sum_i c_i[\sigma_i]$, where σ_i is any p-simplex and $c_i \in \mathbb{Z}$ is a coefficient. Now, we can define the *Boundary Homomorphism* $h_p : C_p \to C_{p-1}$ as a homomorphism defined on any simplex in C [2]. Note that homomorphism is a continuous one-by-one map between two topological space that has also a continuous inverse.

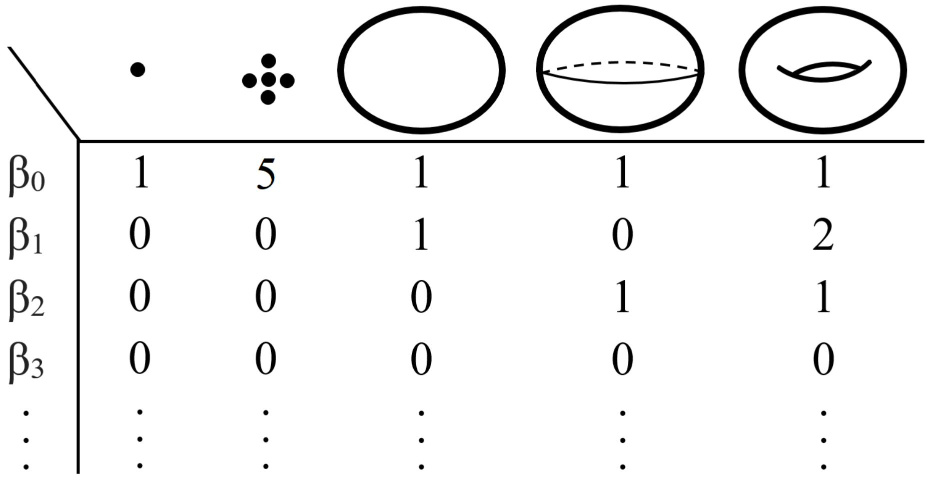
$$h_p[v_0, \ldots, v_p] = \sum_i (-1)^i \; [v_0, \ldots, \hat{v}_i, \ldots, v_p] \tag{2.2}$$

In Equation 2.2, $\hat{v}_i$ denotes the vertex that is deleted from the sequence of vertices.

H_p in Equation 2.1 is the p^{th} *Homology Group* defined as:

$$H_p = Kernel(h_p) \ / \ Image(h_{p+1}) \tag{2.3}$$

In Equation 2.1, $dimC_p$ is equal to the p^{th} *Betti Number.* The i-th Betti number denotes the number of i-dimensional holes in the given simplicial complex. For instance, β_0 counts the connected component (clusters), β_1 denotes the number of 1-D holes and β_2 refers to the number of 2-D holes (voids), etc. Figure 2.2 shows the Betti numbers of some simple shapes.



β_0	1	5	1	1	1
β_1	0	0	1	0	2
β_2	0	0	0	1	1
β_3	0	0	0	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Figure 2.2: Betti numbers of some simple shapes including a single point, a few data points, a single circle, an empty sphere, and an empty donut (torus).

The technique in topological data analysis to refine the topological fingerprints and properties of the data cloud is called *persistent homology* [9, 10, 11]. If we decrease the spatial resolution and then connect any pair of data points which are close enough to each other, some loops may be constructed. If we keep decreasing the resolution, some data records will finally come close enough to each other. So a subset of p different records may become fully connected and assumed to be a p-simplex where there are no loops. A possible requirement to define p-simplex is to have all the pairwise distances between the records in the simplex less than or equal to a fixed range (radius or diameter). The playground of persistent homology is to try all possible radii. When we increase the radius/diameter gradually, many holes (or equivalently loops) will appear and then disappear for each dimension. This process is shown in Figure 2.3. The bottom right of Figure 2.3 shows the persistence diagram–i.e., the diagram that contains the birth radius and the death radius of all the loops (holes) that were born and died through the process in a certain dimension. Persistent diagram is the death vs. birth 2-D plot. Equivalently, the lifetime (birth and death radi) of the loops (holes) can be illustrated with some barcodes in a 1-D diagram, plotted from birth radii to corresponding death radii [12, 5, 3] as in Figure 2.4.

Our discussion on the spatial resolution may imply that the Euclidean distance is used to construct the persistent diagrams or barcodes. But in fact, any other choice of metrics is applicable here. Also, the process of persistent homology as we mentioned, only describes one possible way to threshold the distances referred to as *Vietoris-Rips Filtration* [5]. For the Vietoris-Rips complex, p-simplices are consisting of $p + 1$ nodes where all the pairwise distances are less than or equal to the chosen threshold. An example of a 2-simplex in Rips filtration is illustrated in Figure 2.5. But generally, we may examine some stronger constraints, e.g., all the spheres that are covering the records in the simplex should together have non-empty intersection. This is commonly referred to as *Čech Complex*. An example the Čech Complex is illustrated in Figure 2.6. In another case, working on weighted graphs (as opposed to the data clouds), one may consider thresholding the edge weights and gradually change the weights threshold. The latter example is referred to as *Weight Rank Clique Filtration* [13].

Bubenik in [14] showed that a the topological signatures in data can be also summarized in a real-valued function called *persistent landscape*. Let $\beta(birth, death)$ denote the Betti number of a module at some particular dimension d (d^{th} Betti number) in the resolution interval $(birth\ , death)$. Then the persistent landscape is a real-valued function λ: $\mathbb{N}\times\mathbb{R} \rightarrow \mathbb{R}$ such that $\lambda(n, d) = sup\{radius \geq 0 \mid \beta(d-radius\ ,\ d+radius) \geq n\}$ for any $n \in \mathbb{N}$. Intuitively, if we connect each of those points on the persistence diagram to the bisector line of $\mathbb{R}^2$ via horizontal and vertical lines, and rotate the space by $\pi/4$ clockwise, the result is the landscape function. Persistent Landscape for an equivalent persistence diagram is shown in Figure 2.7. Note that we may calculate or even estimate persistent landscape for different samples of data cloud where the mean of results on different samples still captures the topological signature.

It is worth of mentioning that the algorithm of persistent homology for dimension 0 (connected components) is obviously similar to the idea of distance-based clustering. In fact, both ideas are to analyze the space of distances among data points. However, in clustering, the main question is that which data points are close to each other, while for persistent diagram in dimension 0, the question is how close the data points are to each other. Intuitively, if we assume that the output of the hierarchical clustering [15] is a dendrogram like in Figure 2.8, then we can conclude that clustering deals with horizontal information in the dedrogram (which subsets of the data points are close to each other) and persistence diagram at dimension 0 deals with vertical information—that is at which distance (diameter) the subsets of data are positioned.

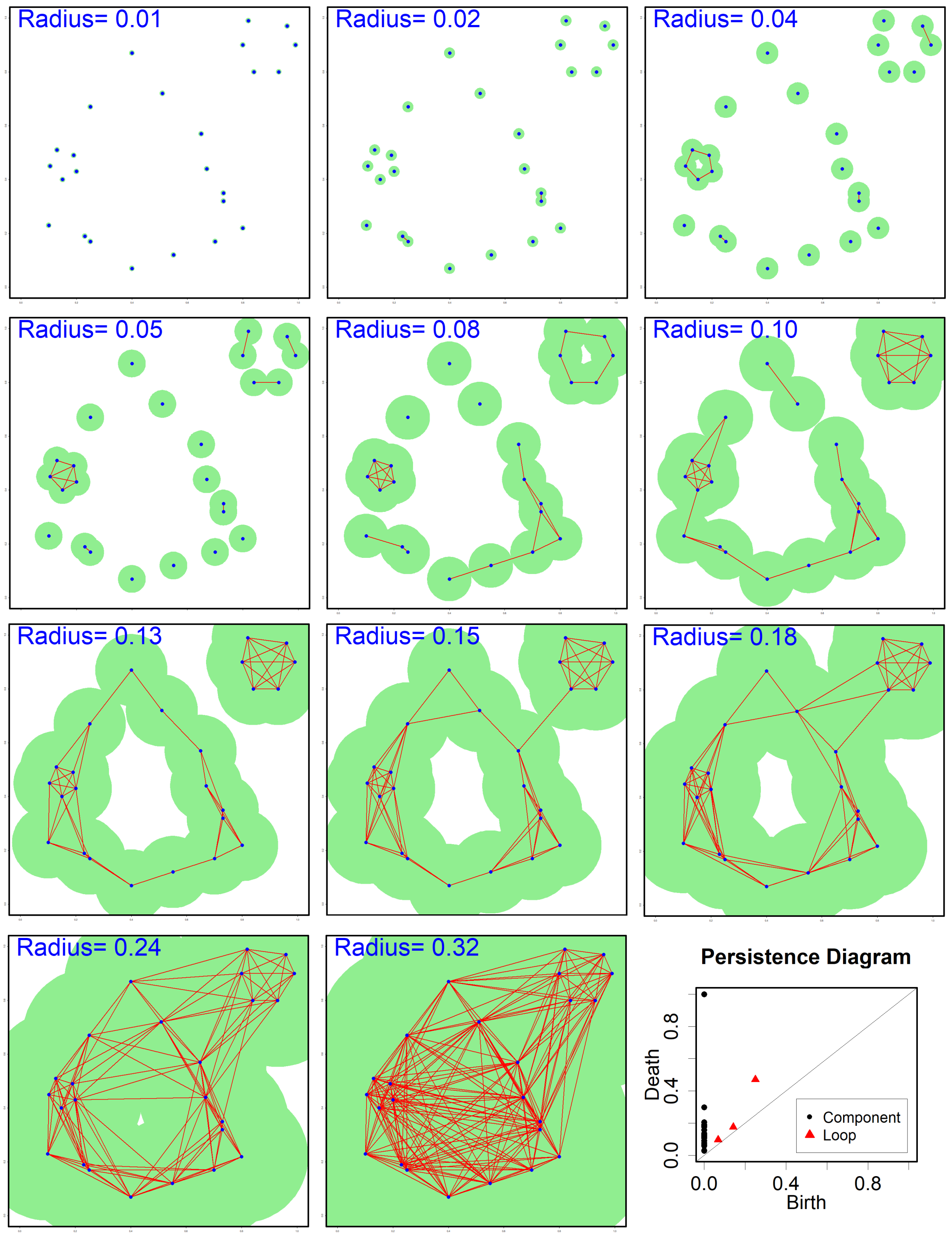


Figure 2.3: An example of using Persistent Homology. Defining an area around each data point with a specific distance (radius) and connect all the points within each others' range, simplices are being distinguished. Increasing the radius gradually, larger simplices will fade in. The persistence diagram represents the summary of the described process.

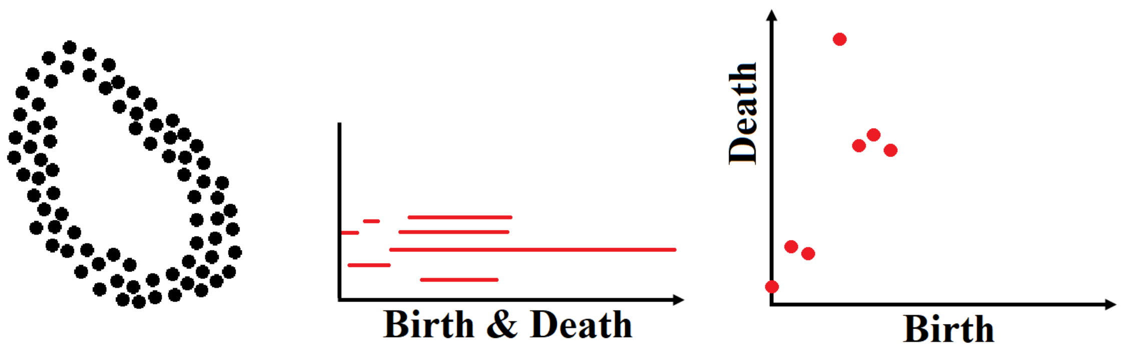


Figure 2.4: A simple example of a data cloud (left), barcodes representation (middle) and its equivalent persistence diagram (right).

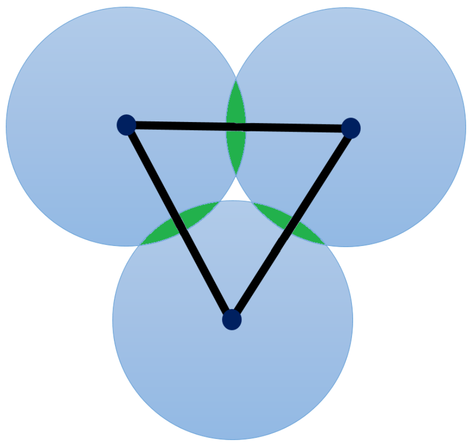


Figure 2.5: A 2-simplex in Rips filtration. In Rips filtration, p-simplices are consisting of $p + 1$ nodes where all the pair-wise distances are less than or equal to the chosen threshold. Here the green regions are satisfying Rips filtration condition.

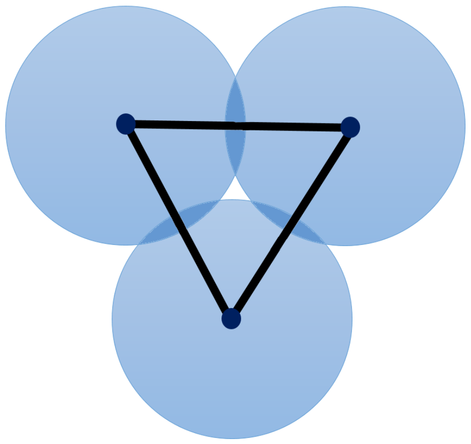


Figure 2.6: A complex that does not satisfy Čech complex condition (right) and a Čech complex (left). In Čech complex, all the spheres that are covering the records in the simplex should together have non-empty intersection. Here the green region satisfies the Čech complex condition.

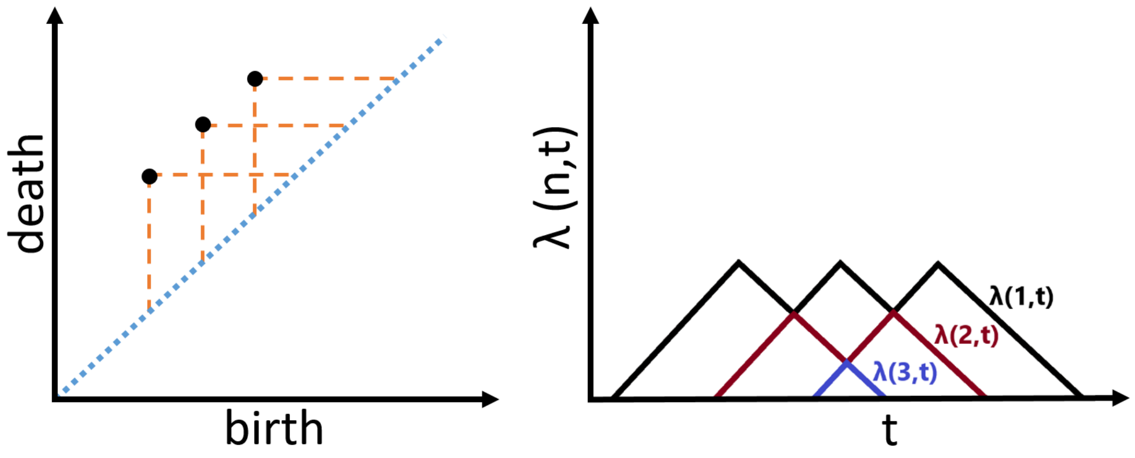


Figure 2.7: Persistent landscape (left) as an equivalent representation of the persistence diagram (right).

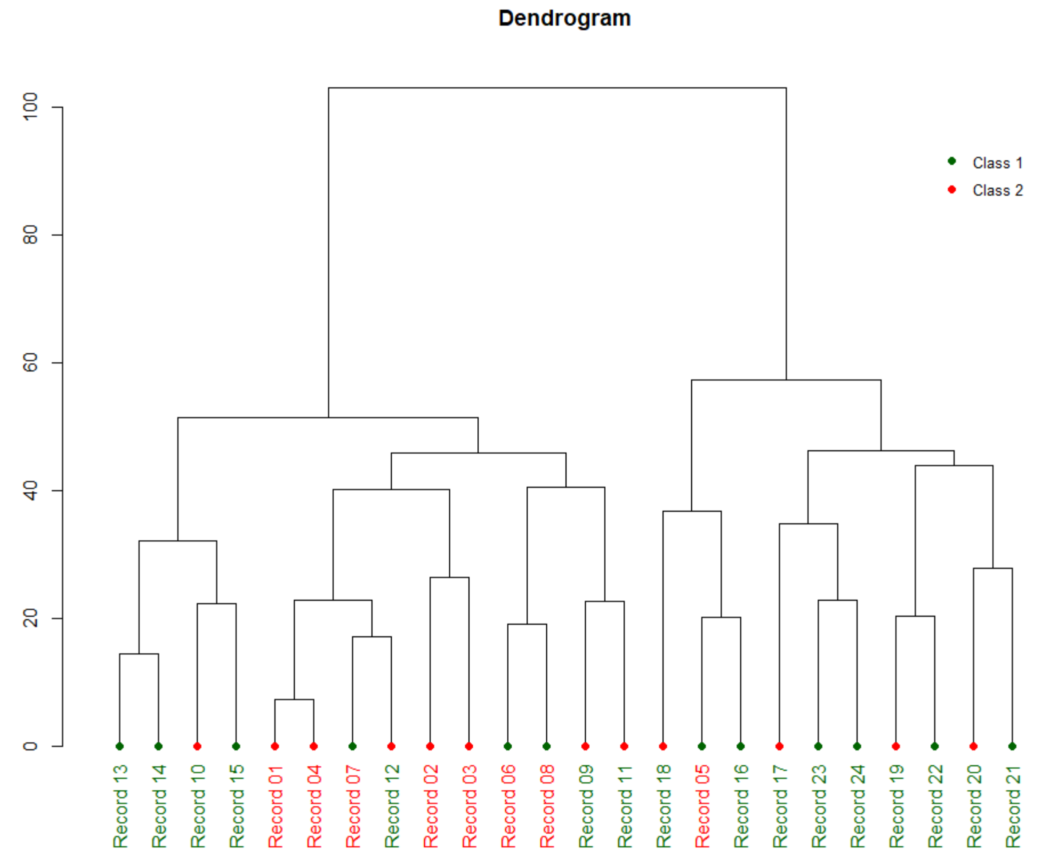


Figure 2.8: A dendrogram visualizes the output of the hierarchical clustering. Clustering mostly deals with horizontal information in the dedrogram (which subsets of the data points are close to each other) while the persistence diagram at dimension 0 analyzes the vertical information (at which distance the subsets of data points are positioned.

CHAPTER 3: Literature Review

Topological Data Analysis is not widely examined in the area of natural language processing and text mining methods. A simple reason would be the non-trivial procedure of defining shapes in the natural language. There have been only some initial efforts to use topological inference for text processing. But more than the literature of text mining, our proposed developments and experiments are inspired by some promising experiments on the applicability of topological inference to the time series analysis [16, 17, 18, 19, 20]. Intuitively, many of the ideas in the application of TDA in time series and signal processing can be borrowed for text processing. In those studies, persistent homology is often used to measure discontinuity in the topological properties of high dimensional time series or the time delay embedding of a one dimensional time series. In some of those experiments, we may easily substitute the sequential signal with some long textual documents. Processing the textual document as a sequence of some different blocks and/or entities may help to analyze the order of appearances of those entities. As suggested by the promising experiments of TDA in time series, we may also utilize it to study the order of entities in natural language. Here, we begin by reviewing some of TDA applications in time-series analysis and then explore the few developments in the application of TDA in text processing.

3.1 TDA in Time Series

Skraba et al. in [21] showed that persistent homology can be used in time-delay embedding models. Time delay embeddings translate a 1-dimensional time series to a d-dimensional time series in which the current value at each time with $(d-1)$ lags coordinate [22, 23]. Skraba et al. developed a framework of analyzing dynamic systems based on topological data analysis that requires almost no prior information of the underlying structure. Instead, a discrete sample of data points is being studied in periodic, quasi-periodic and recurrent systems. The same approach has been considered frequently in the later works on different applications of time series. The authors suggested that clustering d-dimensional delay embedding of a time series in some subspaces of $\mathbb{R}^d$ can easily reveal the recurrent nature of the system on appeared loops and returning paths. The next step is to utilize persistent homology or more precisely the persistence diagram or Betti numbers to measure these loops. Note that these loops in delay-coordinate embedding do not necessarily exist in the samples of the initial time-series. This is one of the reasons that the authors preferred to work on delay embedding $\mathbb{R}^d$ field. Besides, tuning the time delay parameter will result in a more robust model. Based on Vietoris-Rips or any similar filtration, one can construct the persistence diagram of the delay embedding. As a result, a periodic system will end up with the Betti numbers (or equivalently persistence diagram) of a circle. Similarly, a quasi-periodic system with n periods will end up with the persistent Betti number of an n-dimensional torus. For example when $n = 2$ the system will have the Betti numbers of the torus in Figure 2.2. Finally, the persistence diagram of a recurrent system will look like that of a bouquet of circles. Note that in the first two cases, after extraction of persistence diagrams there exist many easy ways to retrieve the time periods explicitly. The persistence diagrams for some of the simple shapes that may appear in time delay embedding space are shown in Figure 3.2 along with their corresponding persistence diagrams.

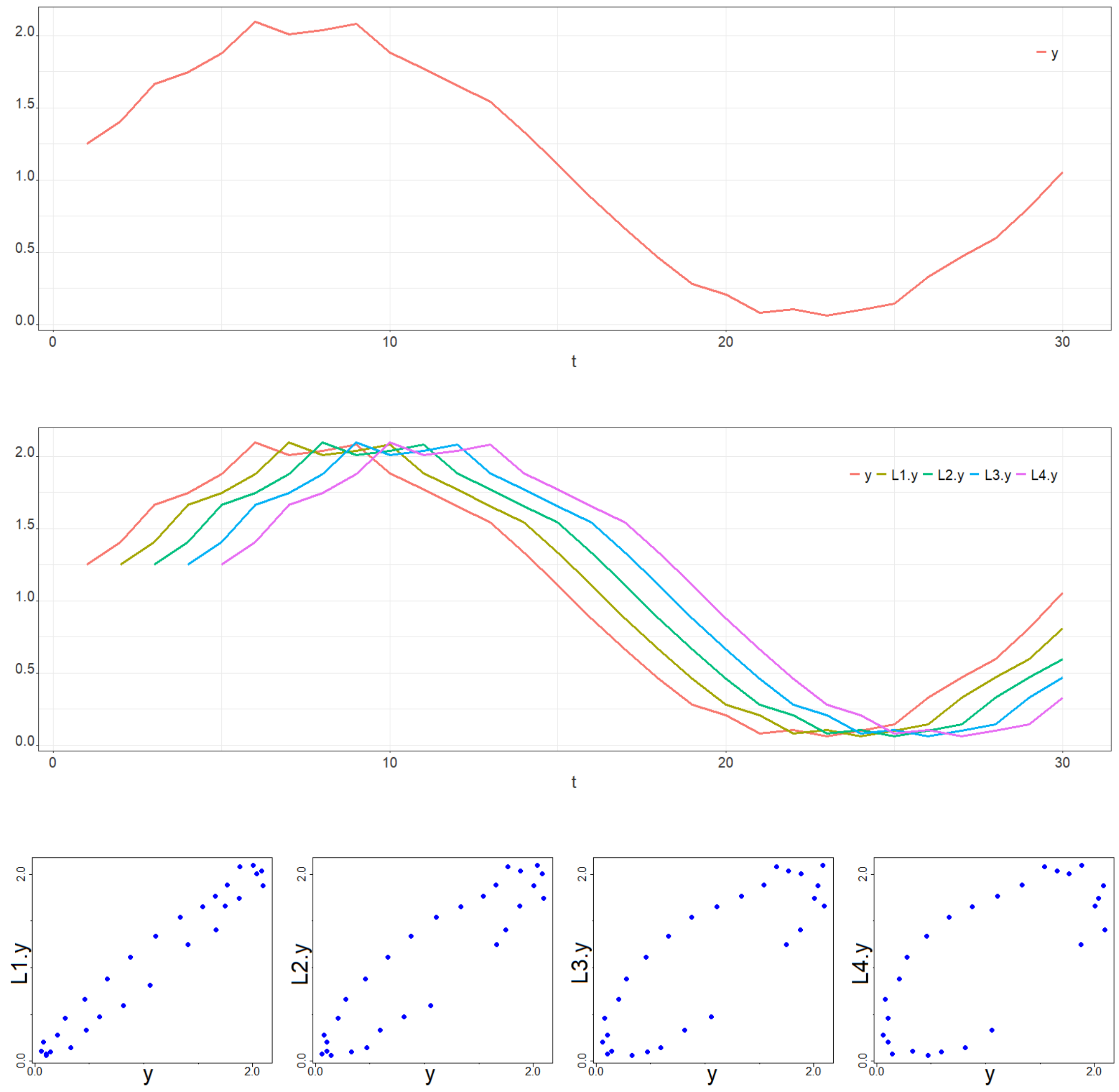


Figure 3.1: An example of time delay embedding in time series: the original 1-dimensional time series (top) is translated to the delay embedding, e.g. with $d = 4$ using $(d-1)$ lags (middle), then we may interpret the d dimensional lags space. Here we illustrates the first 4 lags vs the original signal for simplicity (bottom), but in practice all the lags space are compared in d-dimensional space which is not easily visualizable.

Berwald and Gidea in [24] used persistence diagrams to detect critical transitions

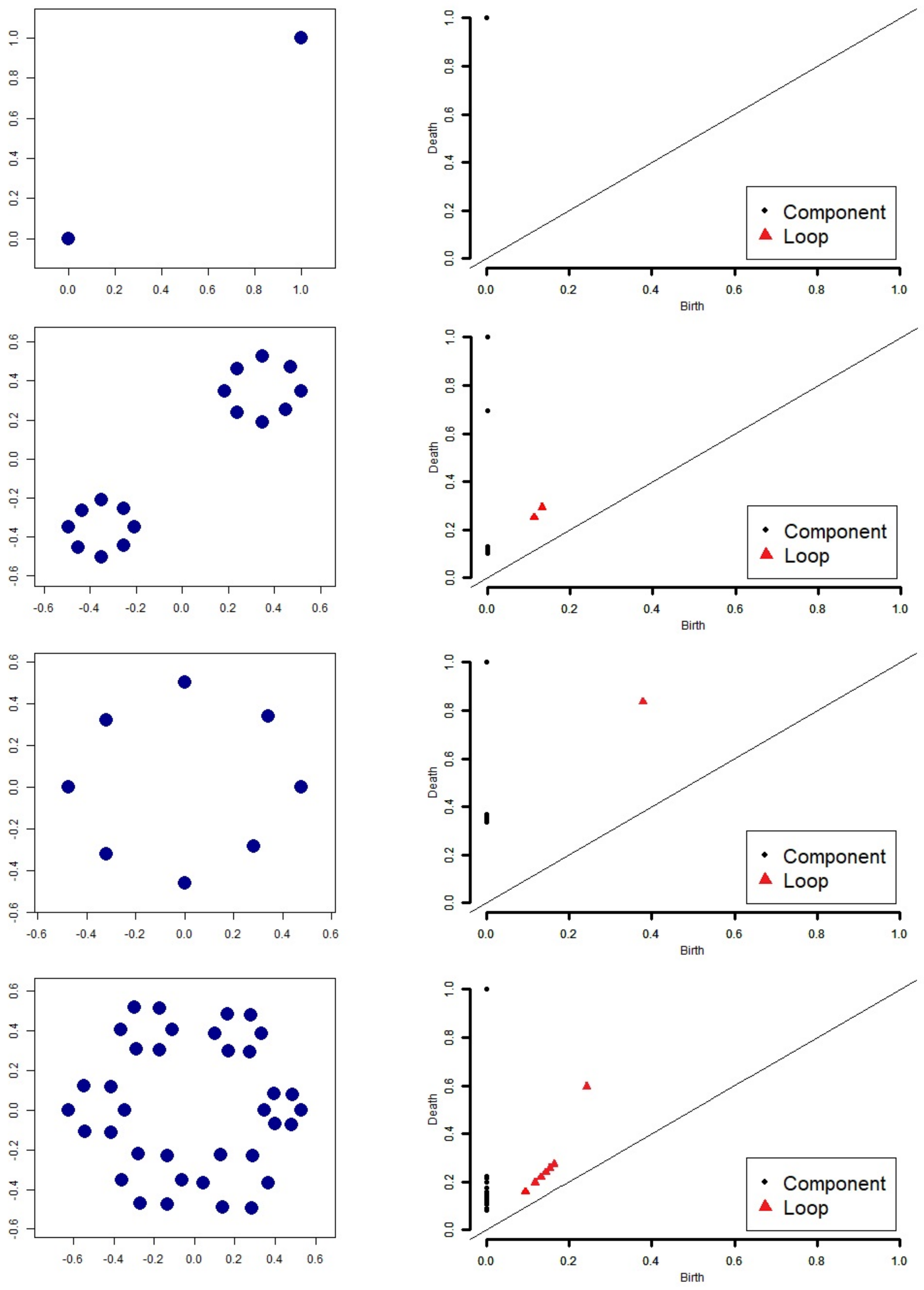


Figure 3.2: Some simple shapes that may appear in time delay embedding space and/or sliding windows (left) and their corresponding persistence diagrams (right).

in genetic regulatory systems that have stochastic characteristics. Based on Vietoris-Rips filtration, the authors built the persistence diagrams of time windows in data and then tried to reveal qualitative changes based on the significant topological difference in consecutive windows. The idea was expanded in [25] where Berwald et al. suggested a novel approach to distinguish different regimes in a time-dependent dynamical system. Constricting the barcodes in dimension 1, the authors used k-means algorithm to cluster the time windows with the most significant bars. Intuitively, if a bar is significantly longer than all the other bars, one may assume that the barcode suggests a periodic regime. There also exist similar conditions to recognize quasi-periodic and recurrent regimes as discussed in [21]. But here the authors feed the barcodes of different time windows to an unsupervised algorithm (e.g., k-means). As a result, local bifurcations are recognizable where the window size is relatively small, or when the cluster tag is changing frequently over time. On the other hand, global bifurcation is detectable when the cluster tag is changing only once or more precisely when the distributions of cluster tags are significantly different before and after a certain time.

Garland et al. in [26] showed how the *Witness Complex* may be used to explore the underlying topology of a noisy sample in dynamical systems. Here the intuition is similar to [25] except the fact that to build the persistence diagrams, the authors utilized *Witness Complexes* [27] instead of Vietoris-Rips complexes. Witness Complexes are often in a smaller size and therefore easier to handle. In a weak Witness complex, a subset (called *landmark*) LM of data cloud DC is chosen. A simplex $Sim = \{l_1, \ldots, l_n\}$ has a *weak witness* $\omega \in P$ if and only if for any $(l^*, r) \in Sim \times (LM \backslash Sim)$ we have $Distance(l^*, \omega) \leq Distance(r, \omega)$. While choosing a landmark subset via some heuristic reduces the computational cost by far, witness complexes will still cover useful topological features. Mittal and Gupta in [28] followed this idea and tried to use persistence diagrams from witness complexes to describe and detect chaos and bifurcations in time series. The authors introduced six different useful features from the persistence diagram including the number of holes, the average lifetime of holes, the maximum diameter of holes and the maximum distance between holes in each dimension. Useful features chosen upon the feature selection training may feed the descriptive or predictive models.

Chazal et al. developed the idea of persistence-based clustering in [29]. In mode-seeking clustering, the local peaks of density function are discovered via some hill-climbing method and are being used as the cluster centers. Since these centers are often unstable, the authors suggested that stable peaks can be found in terms of persistence. While any local peak will present a unique point to the persistence diagram, insignificant peaks (e.g., peaks that appeared due to sampling limitations) are to be eliminated. This is a trivial problem on the persistence diagram. Only the most important peaks can have a long lifetime (i.e., the gap between the birth radius and the death radius in Vietoris-Rips filtration). So, the lifetime of points in the persistence diagram reveals an underlying hierarchy for centroids' importance. Moreover, the persistence diagram can suggest the number of significant points and the efficient number of clusters. Chang et al. in [30] proposed multi-persistent clustering analysis to cluster molecular dynamics simulation data based on scale and density. Similar to [29] the clusters are defined in the terms of persistence, but here the authors measured persistence on a $2D$-space based on scale and density of data points. Pereira and de Mello in [16] also followed the general idea in [29] but more focused on topological properties of each cluster and the relations among them. Their proposed method uses two-dimensional delay embedding of time series. The authors showed that how different types of time and the resulted embedding will result in different barcodes.

Khasawne and Munch in [17, 31, 32, 33] utilized the idea of measuring persistent homology on time delay embedding to evaluate the stability of time-dependent systems in which the main characteristics are defined by Stochastic Delay Differential Equations. They focused on chatter prediction problem [34]. When the delay and the stochastic terms simultaneously exist in the system, it is usually difficult to summarize the behavior and determine the stability, though in special cases some methods combine stochastic calculus and numerical methods to evaluate SDDEs (e.g., in Itô sense). Here the authors used a delay embedding of the signal similar to [21] to construct persistence diagrams and distinguish the system conditions (i.e., equation parameters) under which the system is stable.

Perea and Harer in [18] showed how persistent homology can be utilized to detect the periodic behavior of signals. The authors used 1-dimensional persistence diagrams of sliding window on periodic noisy signals to discover periodicity. They showed how their approach can comparably perform as the state of the art. Perea in [35] extended the theories in [18] to the quasi-periodic functions (i.e., linear summation of periodic functions with irrelevant frequencies). The author proved the way to obtain the optimal choice of time delay and window size for sliding window embedding of such functions and then calculated the upper bound and lower bound for persistent homology of the sliding window.

Predicting the structural breaks and catastrophe stock market crash as early as possible is considered one of the most valuable yet difficult tasks in the financial domain. Gidea in [36] utilized the idea of using persistent homology in [18, 35] to detect the early signs of critical transitions in the financial time series. Here the time series of each stock of an index may depend on some particular condition but on the other hand these indices often respond to mutual information, announcements and regulatory entities. That is how the topology of their correlation graph matters. Considering the time series of multiple stock returns at each instant time as the nodes of a weighted graph, cross-correlations among the nodes are being assumed as the weights of the graph. So at each time there exist a weighted graph representing the correlations of stock returns. To get the persistence diagram for each of the graphs, the author used weight rank clique filtration which thresholds the weights and increase the weights stepwise. Finally, the differences between each persistence diagram and the persistence diagram at the initial time will construct the time series of differences. Here the differences are calculated based on *Wasserstein distance* [37] which measures the minimum cost to map a distribution to another one. Assuming that the persistence diagrams are robust in normal conditions, the time-series of distances can represent and reveal the loss of normal market conditions. The author applied the proposed method to the data set of companies in Dow Jones Industrial Index at time interval including 2007-2008 financial crises and the early signs appeared eight months before the last stock market peak in October 2008. Here the approach is much similar to what was previously done in [38] except the idea of using persistent homology. Nobi et al. in [38] worked on time series of global and local stock indices, built the correlation graph from of these time series in sliding window and got into a sequence of threshold graphs. However here the authors tried to measure the topological change in the sequence of correlation graphs based on the Jaccard index between any two consecutive graphs and focused on degree distribution on the nodes of the graphs that reveal important information regarding network density.

Gidea and Katz in [39] utilized persistent landscape to predict crashes on financial time series. Their method is similar to the general approach in [36]. Here the author used *L^p-norms* (i.e., Minkowski distance of order p from the origin of Euclidean space) of the persistence landscape to detect the signs of a crash. A financial crash often happens after a period of high variance in is market indices and high cross-correlations among stocks. Thus, intuitively the L^p-norms may have a rising trend before the crash. The authors used S&P 500, DJIA, NASDAQ, and Russell 2000 data and analyzed their daily return prior to dotcom crash in 2000 and also 2008 financial crisis. Four indices make a 4-dimensional time series that was analyzed in a sliding window of a fixed size with sliding steps set to one day. Calculating the L^p-norm for each dimension n of the persistent landscape $\lambda(n, x)$ within each window, we will have a vector of L-norms. Finally, the L^p-norm of this vector is a function of time that is used for analysis purposes. As a result, L^1 and L^2 norms illustrated significant rising trends before the crashes.

Sanderson et al. in [40] suggested that persistent homology may capture the difference when the same musical note is played on some different instruments. The authors used persistence diagrams of 2-dimensional delay embedding to distinguish musical instruments. They trained a classifier on a few labeled diagrams of different instruments. The classifier performed much better than a traditional classifier based on the Fast Fourier Transform of time series. Instead of feeding the persistence diagram directly to the classifier, the authors used Persistent Rank Functions– i.e., functions that capture the number and position of points on persistence diagrams [41].

Topological data analysis has been applied to a few other problem statements in time series and dynamical systems analysis. Myers et al. in [42] used persistent homology to study the graph representations of time series derived from dynamical systems. Tymochko et al. in [43] utilized TDA to measure diurnal cycles of hurricanes. Maletić et al. in [19] studied the persistence diagrams of delay embedding in dynamical system similar to [21] and [25], and categorized the expected results on many well-known non-linear systems of chaotic behavior.

Emrani et al. [44] utilized persistent homology for wheeze (Constant Noise) Detection. Considering that the noise frequencies are piece-wise constant, they assumed

Table 3.1: Studies covering TDA in time series and signal processing.

Study	Task/Application
Skraba et al. [21]	Analysis of periodic, quasi-periodic, and recurrent system
Berwald & Gidea [24]	Detecting critical transitions in time series with stochastic characteristics
Berwald et. al [25]	Distinguishing different regimes in time-dependent dynamical systems
Maletić et al. [19]	Categorizing the expected diagrams of well-known chaotic systems
Khasawne & Munch [17, 31, 32, 33]	Stability determination in stochastic systems (chatter prediction)
Garland et al. [26]	Underlying topology of noisy dynamical systems
Mittal & Gupta in [28]	Detecting bifurcations in time series
Chazal et al. [29]	Persistence-based clustering
Chang et al. [30]	Clustering molecular dynamics
Pereira & de Mello [16]	Properties of different time series clusters
Perea & Harer [18]	Analysis of the periodic behavior of signals
Perea [35]	Analysis of quasi-periodic functions
Gidea [36]	Detecting signs of critical financial transitions
Gidea and Katz [39]	Predicting financial crashes
Sanderson et al. [40]	Distinguishing musical instruments
Emrani et al. [44]	Wheeze (Constant Noise) Detection
Khasawneh & Munch [45]	Step detection in periodic signals

that at least 1 persistent barcode in dimension 1 exists in the barcodes of the noise signal. The authors reported a huge difference between barcodes of non-wheeze and wheeze signals. In [45], Khasawneh and Munch used TDA for step detection in periodic signals. Looking at periodic piecewise constant signals, they tried to detect the true steps. Topaz et al. in [46] used persistent homology to study biological aggregations (i.e., dynamical systems defined by a scholar time-series for multiple agents who interact and influence each other. Venkataraman et al. in [47] used persistence diagrams of time delay embedding for human action recognition based on three-dimensional motion capture data. Labeling a few diagrams, they showed how persistent homology can outperform the state of the art in classifying action– e.g., jump, run, sit, dance, or walk. Perea et al. in [48] used persistent homology of a sliding window for periodicity detection in time series of gene expression. Seversky et al. in [49] provided a collection of persistence diagrams from a variety of well-known time series sources that can be used for future studies (e.g., feeding the persistence diagrams to any unsupervised and even sometimes supervised algorithm in traditional machine learning). Stolz et al. in [20] constructed weight rank clique filtration from the time series of functional brain networks and interpreted the resulted persistence landscape.

Some of the studies on the application of TDA and homological persistence in time series and signal processing are organized in Table 3.1.

3.2 The Initial Efforts towards Topological Text Mining

Many studies in text analysis have utilized graphs, usually knowledge graphs for information retrieval, recommender systems, community detection or text summarizations. More specifically, the graph-based approach in natural language processing covers a variety of applications including information retrieval on citation networks [50], paragraph-based text representation using complex networks [51], natural language generation using Knowledge graph [52, 53], plagiarism detection using knowledge graph-based representation [54], knowledge graph-based approach in the cybersecurity domain [55], knowledge graph embedding for query construction [56], graph-based opinion spam detection [57, 58], graph-based semantic representation [59, 60], graph-based semi-supervised learning [61], text summarization using graph theory and entropy [62, 63, 64], synset expansion on translation graph [65], Multilingual knowledge graph embeddings [66], and graph-based knowledge reasoning [67]. However, in most of these studies, the topology of the resulted graphs is left behind, or at least the graphs are not analyzed via persistent homology, as we focus on this study. Thus, there are only a few works that directly relate topological data analysis to the natural language processing.

Wagner et al. in [68] utilized the *flag complex* on the term frequency representation of a corpus to detect and analyze the similarities among different textual records. The focus of the author was on processing efficiency of the homology calculation for high dimensional TF/IDF sparse matrices. The authors suggested a novel approach to use cosine similarity to measure the distance, threshold the distances and gradually change the chosen threshold to retrieve the cliques (i.e., complete sub-graphs) in the graph whose adjacency matrix is the consisting of the cosine distances. The flag complex is defined as a set of all of the clicks in a given graph. In the study, the authors utilized the discrete Morse theory to compress the flag complexes and compute the Betti numbers.

Zhu in [69] suggested a novel approach to apply homological persistence for text analysis and introduces a new topological representation for textual documents. The method is consisting of dividing the text to some number of different text blocks, building a TF/IDF vector for each of the blocks, and then constructing undirected graph whose based on the pairwise cosine distance of different text blocks. In the suggested graph, each node represents one of the text blocks. Thresholding the weights of the graph, the author analyzed the relations among different text blocks. Persistent homology was used to change the threshold and quantify the topological properties of the graph. The author only used β_0 and β_1 to study the connected components and one-dimensional loops. The study showed that the number of loops in the graph can measure the level of tie-back in the text.

The *similarity filtration* that the author used is highly similar to the Rips filtration. But cosine distance of the blocks is measured instead of the Euclidean distance. As the algorithm does not consider the order in which the blocks are arranged in the original text, the author suggested a second method that assumes the edge between subsequent blocks must always exist in the graph, for any given threshold. The modification makes the algorithm consider the order. This second approach *Similarity Filtration with Time Skeleton* (SIFTS) was examined on some nursery rhymes that include lots of repetitive patterns and some other textual documents without repetitive patterns. The author also used SIFTS to distinguish between adolescent writings and child writings, while the effect of document length was controlled via truncating the writings. The author reported that in 87% of child writings and all of the adolescent writings some holes existed. Also the average number of holes for child writings, the original adolescent writings, and truncated adolescent writings were 3.0, 17.6, and 3.9 respectively. This reveals a significant difference between the topological properties of child writings and adolescent writings.

In [70], Doshi and Zadrozny examined SIFTS to detect movie genres on the IMDB data set consisting of movie plots. They concluded that persistent homology may substantially increase the accuracy in text classification and therefore topological inference should be considered a helpful approach for discourse classification tasks.

An unsupervised method to extract key phrases was proposed by Guan et al. in [71]. Their method uses persistent homology for pruning the semantic graph whose nodes are some candidate phrases. Document summarization in the information retrieval context, often begin with many candidate phrases that are finally ranked in a supervised fashion, and top key phrases are being returned. Here the authors built a graph whose nodes are the candidate phrases where nodes are connected if and only if they are sharing a ratio of the same tokens. Afterward the *topological collapse* method [72] removes the dominated nodes. Here a node v_x is said to be dominated by another v_y if all the neighbors of v_x are also in the v_y's neighborhood. A reasonable assumption here is that the domination condition is satisfied, the dominated vertex is fully explained by another vertex and therefore should be considered unnecessary in the graph. The authors examined their method over the NUS corpus and SemEval-2010 data set corpus reporting better performance than the conventional methods in the terms of F1 score recall and precision.

Almgren et al. in [73] studied the applicability of TDA in social network popularity analysis. They used persistent homology to predict the image popularity in the social network, using Word2Vec representations of the captions from some Instagram random images. Having the cosine similarity of the vectors representing the images, the authors utilized Mapper [74] and clustered the set of images. The study revealed a monotonic increase in popularity ratio over different clusters. Later in [75] the authors tried to formulate the problem as images' popularity prediction and concluded that topological inference can outperform prior clustering algorithms (hierarchical clustering and k-means).

There are some other studies trying to apply TDA to text processing. Christianson et al. in [76] analyzed the topological structure of semantic networks in mathematics texts. A topic detection algorithm using Mapper [74] on TF/IDF space was proposed by Torres-Tramón et al. in [77], where the highly connected components among the set of candidate topics are returned as final topics. In [78], Chiang used simplicial complexes to cluster TF/IDF vectors of textual documents. In [79], Zadrozny and Garbayo suggested a sheaf model that distinguishes the disagreements and contradictions in the corpus. Savle and Zadrozny in [80] utilized TDA for text entailment prediction. In addition to the contributions mentioned in Table 3.2, there are many studies utilizing persistent homology in time series and system analysis [16, 18, 17, 19, 26, 20] whose general logic is considerable for embedding based text processing. We organize some of the most related contributions in Table 3.2

Table 3.2: Studies covering TDA in text processing.

Study	Input	Task/Application
Wagner et al. [68]	TF/IDF	Measuring heterogeneity of documents in corpus
Zhu [69]	TF/IDF	Finding repetitions in text (nursery rhymes / writings)
Torres-Tramón et al. [77]	TF	Topic detection in Twitter data
Almgren et al. [73]	Word Embeddings	Image popularity prediction in social media
Doshi & Zadrozny [70]	TF/IDF	Movie Genre Classification on IMDB movie plot data
Savle & Zadrozny [80]	TF/IDF	Text entailment prediction

CHAPTER 4: Methodology

The feasibility of TDA has been examined by many studies in image processing [81, 82, 83], sensor networks [84, 85, 86, 72, 87, 88], system analysis [21, 45, 25, 17], time series and signal processing [24, 18, 35, 49, 39] and ensemble methods [89, 90, 91, 92, 93, 94, 95]. On the other hand, despite a few instances that were mentioned in the previous sections, there exists no general study on the feasibility of TDA in natural language processing.

A wide class of methods for information retrieval text mining and are based on Bag-Of-Words (BOW) models [96, 97, 98, 99]. These models are not considering the original order of the tokens in the text. Adding parts of the parse tree or part-of-Speech may partially, or using ngrams may at least partially address the issue. More importantly, in another family of methods in conventional text processing, there are some ideas to consider the value of token orders in textual documents, e.g., word embeddings representations to the Convolutional Neural Networks (CNNs) [100, 101, 102], Recurrent Neural Networks (RNNs), or Bidirectional Long Short Term Memories (BiLSTMs) [103, 104, 105, 106]. However, that is unclear to what extent these neural network based methods can reflect the actual importance of orders in textual documents. Therefore, there may exist some gaps between the value of the order in the text and the state of the art power of text processing methods. We would consider this as an opportunity for TDA and especially persistent homology to fill the gap. In other words, topological approaches might be helpful to provide a more effective order preserving method for text processing. In a special case, we can utilize TDA to interpret the word embeddings space of textual documents, i.e., going from word embedding to document embedding.

We will begin with a framework to extract topological features from textual documents without using any conventional features such as TF/IDF and word embeddings. We will show how TDA may be utilized in text processing after extracting time series from the textual documents using named entity recognizer (NER).

Second, we will focus on word embeddings. Word embeddings are the more general family of techniques to convert text into time series. Word embedding algorithms like GloVe [107], Word2Vec [108], fastText [109] or ConceptNet Numberbatch [110] translate a textual document to a D-dimensional time series. As described in the literature review, this is a widely used structure of data in TDA literature. Word embedding is an NLP technique in which words or tokens are mapped into a (usually high-dimensional) coordinate system. The more relevant tokens in such coordinate space are placed in closer proximity. The relevance is often defined as the level of common contexts. Such techniques in practice will provide an opportunity to move toward order-preserving algorithms, e.g., using CNNs, RNNs, or BiLSTMs. Simultaneously, these techniques are providing an opportunity to apply topological data analysis for text processing.

Last but not least, we introduce a method to extract some non-conventional topological features from TF/IDF space. We will show how combining the topological features from word embeddings and TF/IDF with the conventional features may lead to a boost in the performance of text processing models. In all of our methods, we focus only on persistent homology in dimension zero (number of clusters) and dimension one (number of loops), due to the practical reasons such as their affordable computational cost.

4.1 Applying TDA to Text without Using Conventional Features

Here we propose a method to examine whether persistent homology is capable of capturing the order of different entities in textual documents. Choosing 75 famous books from the Project Gutenberg written by some famous novelists in the romanticism era of the 19th century, we evaluate our algorithm. We introduce a novel method [111] that utilizes topological data analysis for author prediction solely based on what we define as the graphs of the leading characters in each of the novels. Table 4.1 includes the authors list and also and the frequency of each author.

4.1.1 Extracting Non-Conventional TDA Features from Tagged Entities

Downloading each novel in raw text format from project Gutenberg, we removed the appendices, tables of contents and other metadata that existed in each raw text file. Then we utilized Stanford CoreNLP API as the Named Entity recognizer (NER) [112] to extract the positions of the appearance for each of the characters in each novel, splitting by sentences, tokenizing, and annotating with the tags of named entities, we retrieved all the positions of each token annotated with "Person" named entity tags (e.g., 'Bob', 'Alice', etc.). This way for each book, we got the token indices (word positions in the text) of all the main characters, so the schema of what we processed is like:

$$Book01: \quad [Bob: [15, 170, 210, 1730, \ldots],\ Alice: [92, 270, 920, \ldots],\ \ldots]$$

$$Book02: \quad [Adam: [1, 133, 156, 1127, \ldots],\ Eve: [31, 88, 1520, \ldots],\ \ldots]$$

$$\vdots \qquad\qquad \vdots \qquad\qquad \vdots \qquad\qquad \vdots$$

$$Book75: \quad [Beth: [5, 150, 198, 1025, \ldots],\ Roy: [13, 95, 1023, \ldots],\ \ldots]$$

For the consistency of the process among different novels, we retrieved only ten leading characters in each novel, assuming that most novels include at least ten characters. Also, we assumed that the most frequently appearing characters are the leading characters in each novel. For each novel, we used the set of indices in which each of the characters appears to measure the distance among them (e.g., the distance between 'Alice' and 'Bob' in 'Book01'). Let V and W denote the set of retrieved indices for the characters 'Ch_A' and 'Ch_B', respectively. Without losing the generality, we may assume W has more indices than V, so we retrieve a subset of W having the same size as V. That is V^* in Equation (4.1), where we assume that $n \leq m$.

$$\begin{aligned} V &= (v_1, v_2, \ldots, v_m) \\ W &= (w_1, w_2, \ldots, w_n) \\ W^* &= (w_1^*, w_2^*, \ldots, w_m^*) \end{aligned} \tag{4.1}$$

Our objective in the method of choosing W^* indices is minimizing the pairwise distances between the indices of V and the indices of W^*. Intuitively, the elements of similar positions should be close to each other. The process is formulated in Equation (4.2), where the constraints guarantee that only n unique elements will be retrieved from W.

$$\begin{aligned} & w_p^* = \underset{w_q}{\operatorname{argmin}} \, |w_q - v_p| \; \forall p \in \{1, \ldots, m\} \ , \, q \in \{1, \ldots, n\} \\ & w_p^* \neq w_r^* \ \ \forall r \in \{1, \ldots, (p-1)\} \end{aligned} \tag{4.2}$$

Having the equal sizes of indices for each pair of characters, we may define a distance measure between the character 'Ch_A' and the character 'Ch_B' more easily, as in the Equation (4.3). In the equation, $\tilde{V}$ and $\tilde{W}$ are obtained by normalization of V

and W^*, respectively. Note that the elements in V or W^* are divided by the length of the book—that is the number of tokens in the book. We may also consider using the expanded or compacted representations of $\tilde{V}$ and $\tilde{J}$ in which the elements are raised to the power of n. We will call them $\tilde{V}^{(n)}$ and $\tilde{W}^{(n)}$, respectively. Equation 4.3 defines the distance between the characters, where $WDist_{0.5}$ denotes the Wasserstein distance [37, 113] with the order of 1/2.

$$Dist_p(\text{‘Ch\_A’ , ‘Ch\_A’}) = WDist_{0.5}(\tilde{V}^{(1+p)}, \tilde{W}^{(1+p)}) \tag{4.3}$$

Note that Wasserstein distance is in fact cost of mapping one distribution to another distribution. Intuitively, working on two diagrams, it measures the movements of dots in one diagrams to get into the second diagram, as illustrated in Figure 4.1.

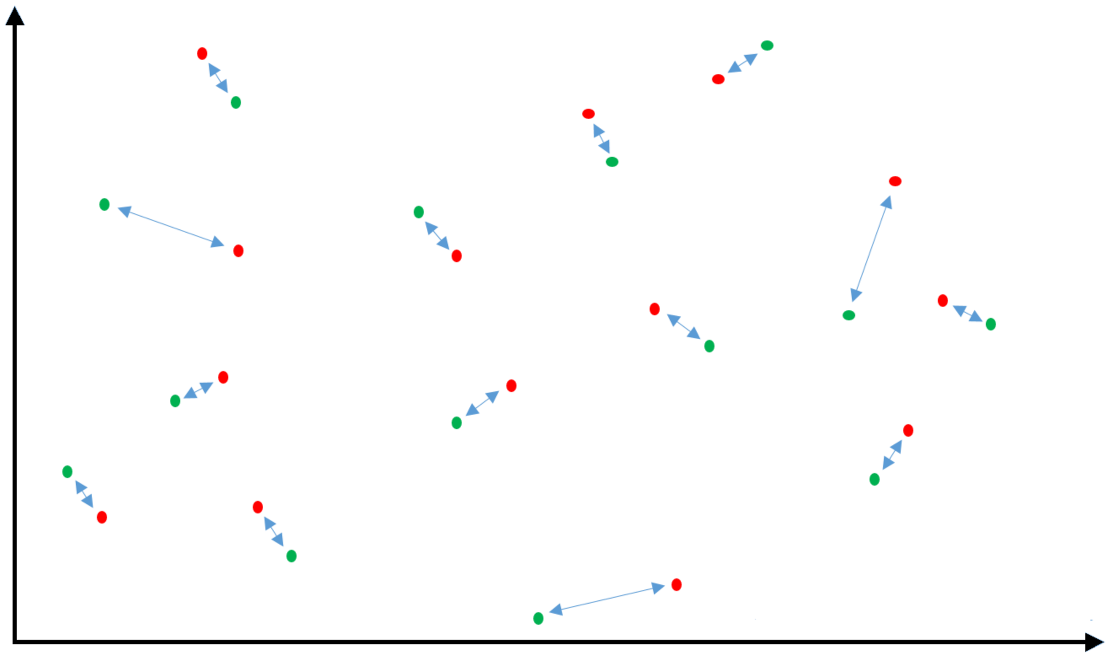


Figure 4.1: Wasserstein distance measures the movements of dots in one diagrams (green dots) to get into the second diagram (red dots).

In Equation 4.3, for $p = 0$ the defined distance is measuring the difference of the original indices vectors ($\tilde{V}$ and $\tilde{W}$. We use the Wasserstein distance of the order 1/2 to push the distance measure being more sensitive where the corresponding indices in the two vectors are closer. The main intuition here is that if we try different values of p, we are in fact more focusing on characters that are co-appearing at the initial chapters of the novel ($p > 0$) or some close characters that are co-appearing at the final chapters of the books ($p < 0$). These two scenarios will add the degrees of freedom and will consequently increase our chance of retrieving the geometrical/topological style of different authors.

Having the pairwise distance between the leading characters in each novel, we used Rips filtration [114, 115, 116] to construct a persistence diagram representation of each Book. For each of the novels, we tried three different values of p ($p = 0, \ +\epsilon \ and \ -\epsilon$) to cover the three different scenarios discussed before and retrieve more homological features. Using $\epsilon = 1/10$, for each novel we built three different diagrams. TDA package [117] in R [118, 119] was utilized for the implementation of the persistence diagrams and and for quantification over the constructed diagrams. Figure 4.2 illustrates some instances of the persistence diagrams for $p = 0$ case.

Assuming that we have made the persistence diagrams of all the given books, we can measure the pairwise distance among the books to quantify the geometric similarities and differences among different books. This would reveal whether the authors are repeating any topological characteristics in their books. In other words, we are interested to know whether (1) different novels by the same author have similar persistence diagrams and (2) different authors usually have significantly different persistence diagrams. We utilized the Wasserstein distance (order = 1) to quantify the differences and compare the diagrams and dimensions zero and one. Equation 4.4 formulates the overall distance for each pair of books. Since each novel, three diagrams are constructed ($p = 0, \ +\epsilon \ and \ -\epsilon$) and each diagram is coming in two

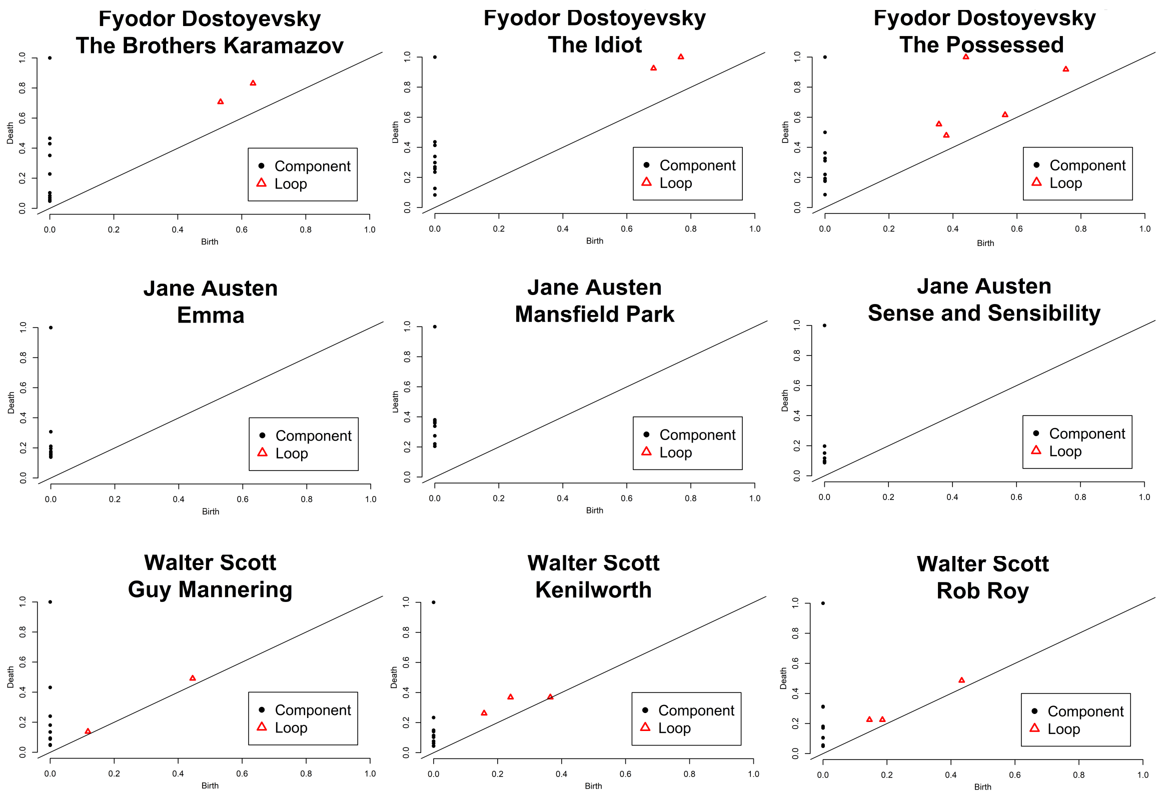


Figure 4.2: Some examples of the persistence diagram constructed from the graph of the leading characters in different books.

different dimensions (dimension zero and dimension 1), we need two combine six different scores for each pair of books to provide the final difference score. In Equation 4.4, $PerD_p^0$ only represents the clusters (connected components) and $PerD_p^1$ represents the holes (loops in dimension 1). Here B and B' are any two given books and $WDist$ denotes Wasserstein distance (order=1).

$$\begin{aligned} Dist_p(B\ ,\ B') &= WDist\{PerD_p^0(B), PerD_p^0(B')\} \\ &\quad +\ WDist\{PerD_p^1(B), PerD_p^1(B')\} \\ Dist(B\ ,\ B') &= \{\sum_{p\in\{-\epsilon,0,+\epsilon\}} Dist_p(B\ ,\ B')^2\}^{\frac{1}{2}} \end{aligned} \tag{4.4}$$

Our method is summarized in Algorithm 1.

4.1.2 Results of TDA on Tagged Entities

Our method of evaluation was consisting of running We used k Nearest Neighbors supervised method in cross validation mode (using ten folds), where the independent variables (predictors) were the distances among different books and the dependent variable (prediction target) was the author of each book. Since we got different numbers of books by each author, for binary author classification, we used balanced training samples. For example when classifying author A against author A' where author A has m books and author A' has n books, assuming $n \geq m$ (without loss of generality), we randomly selected m books from author A to get a balances set. Also, to cover all possible combinations, for each pair of authors, we iterated 250 times through the balanced records and each time measured the performance of the cross validation. Based on our manual grid search we chose $k = 5$ for the kNN model. In our modified kNN model, each of the neighbors has a specific vote value inversely

Algorithm 1 Compare Books Using Topological Features from Name Entities

```
Input: plain textual documents:
    A list $Book = [book(1), \ldots, book(n)]$ where book(i) is the plain text of i-th document.
Output: A matrix $Distance_{n\times n}$ where $Distance[i, j]$ is the distance between book(i) and book(j).
 1: for $i = 1$ to $n$ do
 2:     Retrieve all name entities tagged as PERSON in book(i).
 3:     Keep 10 most frequent name entities in book(i).
        Save their corresponding indices in $Person_1(i), \ldots, Person_{10}(i)$.
 4:     for $v, w = 1$ to 10 do
 5:         Compute distance between $Person_v(i)$ and $Person_w(i)$ under parameter p—
            that is $Dist_p\{Person_v(i), Person_w(i)\}$ for $p \in \{-\epsilon, 0, +\epsilon\}$ using Equation 4.3.
 6:     end for
        Using the adjacency matrix $Dist_p$ make the persistence diagram $PD_p(i)$ of
        book(i) for $p \in \{-\epsilon, 0, +\epsilon\}$.
 7: end for
 8: for $i, j = 1$ to $n$ do
 9:     Calculate the distance between the i-th and the j-th books, $Distance[i, j]$ based
        on their persistence diagrams using Equation 4.4.
10: end for
11: return Distance matrix $Distance_{n\times n}$
```

proportional to its squared distance.

While every writer or author may change her style over the time, it still sounds reasonable to assume that the writers usually their specific style in at least some different works. That is why our topological method might be capable of providing some informative neighbors for the kNN model. However, what we call style here, only denotes the pairwise proximity of different leading characters. The performance of the pairwise (one-vs-one) classifications is shown in Table 4.1 in terms of accuracy. Note that the total number of predictions was around 69k and a mean accuracy of 77.0% was achieved. Table 4.1 suggests that accuracy can be very high in some cases (pairs of authors) and much lower in some other cases. For instance, distinguishing Jane Austen's books from Fyodor Dostoevsky's looks much easier than distinguishing Fyodor Dostoevsky's works from Émile Zola's or Walter Scott's as also Figure 4.2 reveals. Sometimes these variations can be due to the limitations of our homological methods but note that the real similarity of different authors may also confuse the classifier.

Table 4.1: Cross validation performance in one-vs-one classification. Numbers inside parentheses denote the number of processed books by each author. While the classifier can be highly accurate in some cases, in some other cases, the accuracy can be as low as 55%. In average, the achieved accuracy is 77%.

	W.Scott (18)	**M. Twain (8)**	**J. Austen (6)**	**F.Dostoevsky (8)**	**É. Zola (18)**	**C. Dickens (17)**
W. Scott	73.9	83.3	55.8	94.7	68.5	-
M. Twain	74.6	68.8	73.3	82.9	-	68.5
J. Austen	100	64.2	90.2	-	82.9	94.7
F. Dostoevsky	72.2	65.0	-	90.2	73.3	55.8
É. Zola	87.0	-	65.0	64.2	68.8	83.3
C. Dickens	-	87	72.2	100	74.6	73.9
AVG	75.2	73.6	86.4	71.3	73.7	81.5

In our study, for the initial named entity recognition step, we skipped using Co-reference Resolution [120, 95]. The reason is that currently, co-reference resolution tools are not useful enough for our proposed method. We may acknowledge that this way many of the possible indices are being missed by the NER, especially when the character is indirectly mentioned or implied with some pronouns. However, we may assume that for most of the missed indices, the corresponding leading character (person) in the story is directly mentioned a few lines above or bellow the missed pronoun. For the purpose of this study, retrieving the most reliable named entities is much more important than covering a higher number of named entities. That is why is skipped using co-reference resolution. Note that missing some indices most probably only affects the geometry in the graph of the leading characters, while the topology of the graph is not significantly affected. On the other hand, a high portion of false-positive indices not only change the geometry but can also lead to a critical change in the topology in the graph of the leading characters and subsequently, the topological fingerprint of the author may fade.

The main scope of the proposed study is to examine the existence of homological fingerprints in the text, especially in long writings. Choosing some publicly available novels as the symbolic instances of long textual documents, we examined our method. We still need to investigate if/how the results are extendable to some more realistic problem statements, e.g., the general application of text or topic classification. The other matter of concern would be the cost of computation, especially when dealing with tons of textual documents. Notably, while our results are promising to show the existence of the homological features, the performance is not high enough to assume that we can apply the proposed framework for many real word applications, unless we use the topological features in addition to some conventional text representations such as TF/IDF vectors or word embeddings.

The table of the results reveals that the homological fingerprints of the authors exist in their works. As the main contribution, we analyzed the special names (the leading characters through the books), made the graphs of proximity among the leading characters, and finally used those graphs to extract the homological features using persistent homology. One of the main properties of the framework we introduced is the robustness toward the translation and insensitivity to the document size.

4.2 A Topological Algorithm Using Word Embeddings

In our algorithm for topological inference of the embedding Space, we utilize word embedding and persistent homology. The input is the textual document and the output is a topological representation of the same document. Later we may use these representations for text classification, clustering, etc.

4.2.1 Main Steps in TDA Algorithm on Word Embeddings

Step-by-step specifications of the topological method on the embedding space are explained is this section.

Pre-processing: Like any other text mining method, standard pre-processing possibly including lemmatization, removing stop words and if necessary lowercasing will be applied to the text. Also, there might be some specific pre-processing tasks that are inspired by TDA algorithms.

$$\underset{T\times 300}{\mathfrak{Genesis}} = \begin{array}{r|ccccc|} & \text{d1} & \text{d2} & \text{d3} & \dots & \text{d300} \\ \mathfrak{in} & 0.122 & 0.156 & 0.046 & \dots & -0.034 \\ \mathfrak{the} & 0.124 & 0.167 & 0.033 & \dots & -0.026 \\ \mathfrak{beginning} & 0.118 & 0.082 & 0.009 & \dots & 0.010 \\ \mathfrak{god} & 0.053 & 0.040 & -0.016 & \dots & 0.134 \\ \mathfrak{created} & 0.110 & 0.035 & -0.003 & \dots & -0.029 \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ \mathfrak{coffin} & 0.035 & 0.019 & 0.110 & \dots & 0.025 \\ \mathfrak{in} & 0.122 & 0.156 & 0.046 & \dots & -0.034 \\ \mathfrak{egypt} & -0.094 & 0.043 & 0.014 & \dots & -0.013 \end{array}$$

Figure 4.3: Word embedding representation for the Book of Genesis (King James Version) based on GloVe pre-trained vectors.

Word Embedding Representation: In a document of size T, replacing each token with its embedding vector of size D will result in a matrix Ψ of size $T \times D$.

This matrix can naturally be viewed as a D-dimensional time series that represents the document. More precisely, each column Ψ_d ($d = 1, \ldots, D$) will represent the document in dimension d of the embedding, as shown in Equation 4.5. An example of such representations for the Book of Genesis (King James Version) based on GloVe pre-trained vectors is illustrated in Figure 4.3.

$$\Psi_{T\times D} = \begin{bmatrix} \psi_1^1 & \psi_2^1 & \ldots & \psi_d^1 & \ldots & \psi_D^1 \\ \psi_2^2 & \psi_2^2 & \ldots & \psi_d^2 & \ldots & \psi_D^2 \\ \vdots & \vdots & & \vdots & & \vdots \\ \psi_1^t & \psi_2^t & \ldots & \psi_d^t & \ldots & \psi_D^t \\ \vdots & \vdots & & \vdots & & \vdots \\ \psi_1^T & \psi_2^T & \ldots & \psi_d^T & \ldots & \psi_D^T \end{bmatrix} \qquad (4.5)$$

$$1 \leq t \leq T$$

$$1 \leq d \leq D$$

Aggregation on Sliding Window: One of the easiest and potentially most efficient ways of such smoothing is to replace each element $\Psi_{t,d}$ ($t = 1, \ldots, T$) in Ψ_d with a local average in its neighborhood. Equivalently, we may describe it by taking the summation in the sliding window of size ω, where $c = (\omega - 1)/2$ so the smoothed vector is

$$\tilde{\Psi}_{t,d} = \Psi_{t-c,d} + \Psi_{t-c+1,d} + \cdots + \Psi_{t,d} + \cdots + \Psi_{t+c,d}$$

and $\tilde{\Psi}_{(T-\omega+1)\times D}$ is the smoothed D-dimensional time series that represents the document. For long documents this is almost the same as using a smoother matrix $\Pi_{T\times T}$, i.e.,

$$\tilde{\Psi}_{T\times D} = \Pi_{T\times T}\Psi_{T\times D}$$

where Π is a tridiagonal (for $\omega = 3$), pentadiagonal (for $\omega = 5$) or heptadiagonal (for $\omega = 7$) binary matrix. The only difference is that in the latter definition, no value of time series will be dropped, so the result is only slightly different in size, assuming $T >> \omega$. Note that we can also use exponential weights in summation of elements in the sliding window instead of simply adding them up. In one of our experiments, we tried the exponential form of:

$$\begin{aligned}\tilde{\Psi}_{t,d} = \Psi_{t,d} & \\ & + \frac{1}{8}\Psi_{t-3,d} + \frac{1}{4}\Psi_{t-2,d} + \frac{1}{2}\Psi_{t-1,d} \\ & + \frac{1}{2}\Psi_{t+1,d} + \frac{1}{4}\Psi_{t+2,d} + \frac{1}{8}\Psi_{t+3,d}\end{aligned}$$

Computing Distances: Assume that in a documents, some of the embedding dimensions and the relation among different dimensions are carrying some information regarding the document. Such information could be revealed in a coordinate system, where each embedding dimension is represented by a data point, and the distance between two data points (embedding dimensions) represents their relation. A possible choice of distance is formulated in Equation 4.6.

$$\begin{aligned}\vartheta(\tilde{\Psi}_i, \tilde{\Psi}_j) : &= \sqrt{\mathbb{E}[\tilde{\Psi}_i^2]\mathbb{E}[\tilde{\Psi}_j^2]} - \mathbb{E}[\tilde{\Psi}_i\tilde{\Psi}_j] \\ &= \frac{1}{T}\left\|\tilde{\Psi}_i\right\|\left\|\tilde{\Psi}_j\right\| - \frac{1}{T}\tilde{\Psi}_i^T\tilde{\Psi}_j \\ &= \frac{1}{T}\left\|\tilde{\Psi}_i\right\|\left\|\tilde{\Psi}_j\right\| \{1 - CosSim(\tilde{\Psi}_i, \tilde{\Psi}_j)\}\end{aligned} \tag{4.6}$$

The intuition behind the way of defining distance in Equation 4.6 is to (1) consider the relation between $\tilde{\Psi}_i$ and $\tilde{\Psi}_j$ via Cosine similarity, (2) distinguish significant embedding dimensions (the term $\left\|\tilde{\Psi}_i\right\|\left\|\tilde{\Psi}_j\right\|$ will do this), and (3) make the distance almost non-sensitive to the size of document via term $1/T$. Note that Equation 4.6 can be replaced by any other definition satisfying these three conditions. Aggregation on sliding window along with a distance formula like Equation 4.6 guarantee that the order of the tokens in documents is considered in the final distance matrix Φ, defined by

$$\Theta_{D\times D} = [\vartheta(\tilde{X}_i, \tilde{X}_j)] \quad ; \quad i, j = 1, \ldots, D.$$

For simplicity let's fix $\omega = 5$, so remembering that each column of $\tilde{\Psi}$ as a simple time series is a function of time (the index of word/token in the document),

$$\tilde{\Psi}_i = \tilde{\Psi}_i(t) = \Psi_i(t-2) + \Psi_i(t-1) + \cdots + \Psi_i(t+2)$$

and assuming that the length of document T is large enough ($T >> 1$) we have

$$\mathbb{E}[\Psi_i(t+s)\Psi_j(t+s)] \to \mathbb{E}[\Psi_i(t)\Psi_j(t)]$$

as $T \to \infty$, so

$$\mathbb{E}[\Psi_i(t+s)\Psi_j(t+s)] \approx \mathbb{E}[\Psi_i(t)\Psi_j(t)]$$

since shifting the time index will only exclude $|s|$ elements from the beginning or the end of time series and its effect is negligible when $T >> |s|$. It is easy to show that

$$\begin{aligned}
\mathbb{E}[\tilde{\Psi}_i\tilde{\Psi}_j] &= \mathbb{E}[\tilde{\Psi}_i(t)\tilde{\Psi}_j(t)] \\
&\approx 5\mathbb{E}[\Psi_i(t)\Psi_j(t)] \\
&+ 4\mathbb{E}[\Psi_i(t-1)\Psi_j(t)] + 4\mathbb{E}[\Psi_i(t)\Psi_j(t-1)] \\
&+ 3\mathbb{E}[\Psi_i(t-2)\Psi_j(t)] + 3\mathbb{E}[\Psi_i(t)\Psi_j(t-2)] \\
&+ 2\mathbb{E}[\Psi_i(t-3)\Psi_j(t)] + 2\mathbb{E}[\Psi_i(t)\Psi_j(t-3)] \\
&+ 1\mathbb{E}[\Psi_i(t-4)\Psi_j(t)] + 1\mathbb{E}[\Psi_i(t)\Psi_j(t-4)]
\end{aligned}$$

and similarly, in a general form for any window size ω, Equation 4.7 holds.

$$\begin{aligned}
&\mathbb{E}[\tilde{\Psi}_i\tilde{\Psi}_j] = \mathbb{E}[\tilde{\Psi}_i(t)\tilde{\Psi}_j(t)] \\
&\quad \to w\mathbb{E}[\Psi_i(t)\Psi_j(t)] \\
&\quad + (\omega-1)\mathbb{E}[\Psi_i(t-1)\Psi_j(t)] + (\omega-1)\mathbb{E}[\Psi_i(t)\Psi_j(t-1)] \\
&\quad + (\omega-2)\mathbb{E}[\Psi_i(t-2)\Psi_j(t)] + (\omega-2)\mathbb{E}[\Psi_i(t)\Psi_j(t-2)] \\
&\qquad\quad \vdots \qquad\qquad \vdots \\
&\quad + 1\ \mathbb{E}[X_i(t-\omega+1)X_j(t)] + 1\ \mathbb{E}[X_i(t)X_j(t-\omega+1)] \\
&\text{as } T \to \infty
\end{aligned} \tag{4.7}$$

Such coefficients will guarantee that the order is considered in the final distance matrix θ. It means that each embedding dimension for each token in the text is being compared with all the other embedding dimensions in the same token, a few tokens before that, and a few tokens after that, though these comparisons will have different weights, as illustrated in Figure 4.4. Note that similar equations can be easily derived for correlation-based and covariance-based distances. In any case, the distance matrix is sensitive to the window size ω, or more generally to the smoothing algorithm. For instance, using exponential smoothing will result in geometric sequence of coefficients instead of arithmetic sequence of coefficients in Equation 4.7 (i.e., ω, $\omega - 1$, …,1). Regarding using the sliding window, the choice of ω is a trade-off between increasing the captured information on one side and decreasing the noise on the other side.

Applying Persistent Homology: Having the distance matrix Θ for each document, a persistence diagram $PD(\Theta)$ can be constructed for topological dimension[1]

[1]These dimensions should not be mistaken with embedding dimensions.

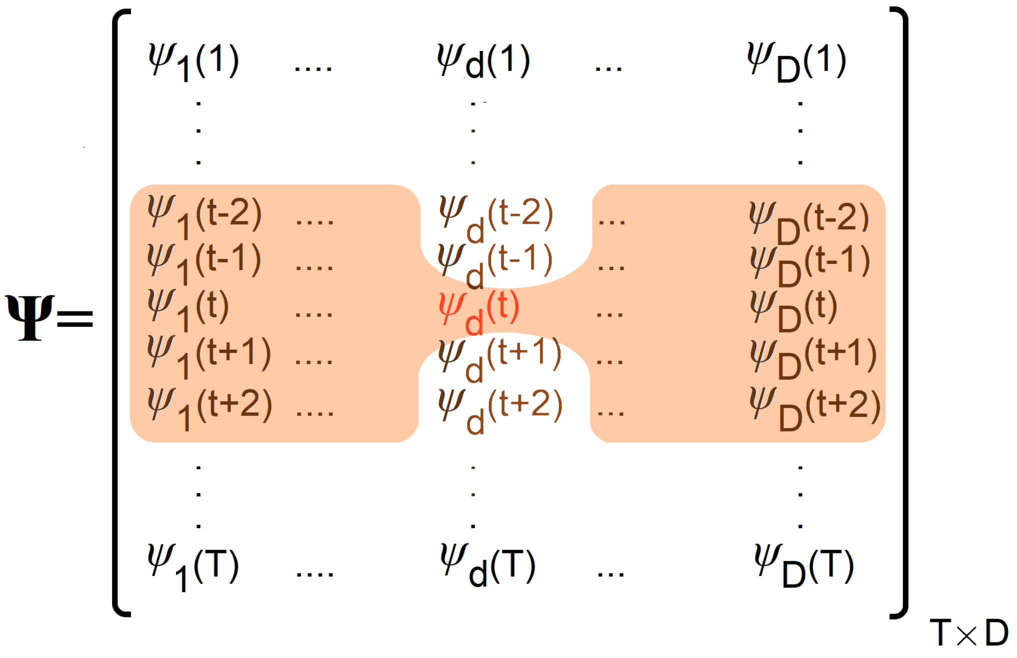

Ψ=
ψ1(1) .... ψd(1) ... ψD(1)
ψ1(t-2) .... ψd(t-2) ... ψD(t-2)
ψ1(t-1) .... ψd(t-1) ... ψD(t-1)
ψ1(t) .... ψd(t) ... ψD(t)
ψ1(t+1) .... ψd(t+1) ... ψD(t+1)
ψ1(t+2) .... ψd(t+2) ... ψD(t+2)
ψ1(T) .... ψd(T) ... ψD(T)
T×D

Figure 4.4: Working on the smoothed matrix, during distance calculation step, each embedding dimension for each token in the text is being compared with all the other embedding dimensions in the same token, a few tokens before that, and a few tokens after that. Note that the comparisons will have different weights that depends on the smoothing function.

0 (number of clusters) and dimension 1 (number of loops) denoted by $PD_0(\Theta)$ and $PD_1(\Theta)$ respectively. However, this persistence diagram alone is not very useful as the representation of the document.

In our previous algorithm [111] the resulted graph of the proximity of the leading characters (persons) in each book, and therefore the distance matrix was not annotated nor was needed to be annotated, since we had designed the algorithm to deal with the main characters whatever their names are. In other word, to capture the topological signature of a novelist, it did not matter whether the names of the main characters are shifted. But here, dealing with time-series in different dimensions, the order of embedding dimensions is meaningful, since different embedding dimensions have different roles in representing document. Therefore, feeding the time series to the persistent homology algorithm is meaningless, unless we somehow manage to distinguish different dimensions. If we fail to address the issue, the embedding space that persistent homology perceives is not annotated and the map of different embedding dimensions will be lost as shown in Figure 4.5.

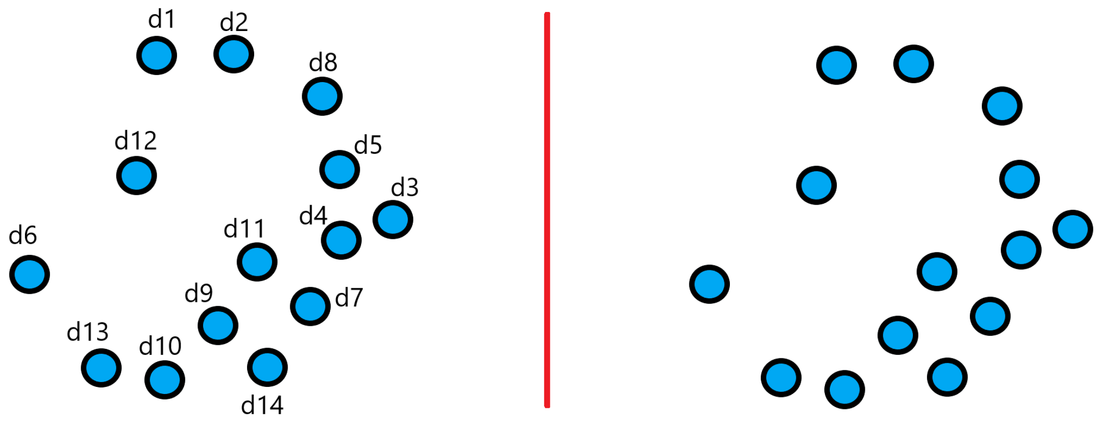


Figure 4.5: The embeddings distance space annotated by the embeddings indices (left) and the embeddings distance space that persistent homology may perceive.

One intuitive way is comparing the persistence diagram with and without each embedding dimension. In other word we can measure the change in persistence diagram when we exclude one embedding dimension. We measure the sensitivity of the persistence diagram generated by Ripser [121, 122] to each embedding dimension to use it later as a measure of the sensitivity of the document itself to each embedding dimension. This way the document will be represented in an array of size array of D based on $PD_0(\Theta)$ and another array of size D based on $PD_1(\Theta)$, as formulated in Equation 4.8, where W is any measure of distance between two persistence diagram, e.g, Wasserstein distance.

$$V_0(d) = \Psi(\ PD_0(\Theta)\ \ ,\ \ PD_0(\Theta \setminus d)\ )\quad ;\quad d = 1, \ldots, D$$
$$V_1(d) = \Psi(\ PD_1(\Theta)\ \ ,\ \ PD_1(\Theta \setminus d)\ )\quad ;\quad d = 1, \ldots, D \tag{4.8}$$

The steps are summarized in 2 and A block diagram of the algorithm is shown in Figure 4.6.

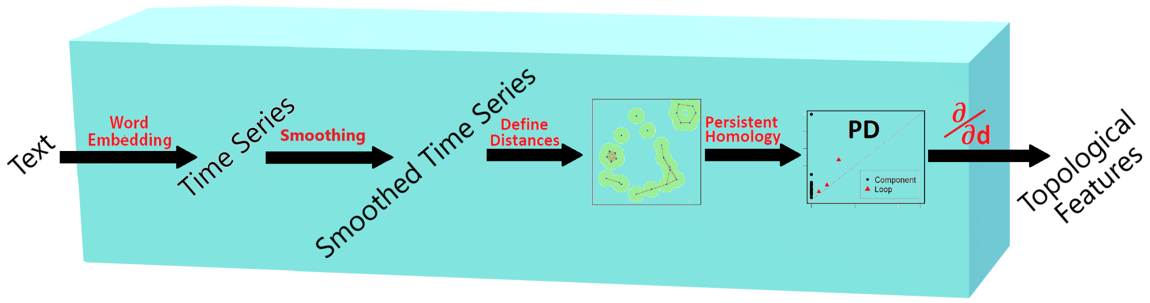


Figure 4.6: A block diagram of the topological algorithm on the embedding space. Word Embedding representations are aggregated over sliding windows. Then defining the distance between pairs of embedding dimensions, persistence diagram can represent the text. Finally, the topological features are the sensitivity of the persistence diagram with respect to different dimensions.

4.2.2 Data Specification to Apply TDA on Word Embeddings

To examine our topological algorithm on the word embeddings space, we use the following data sets and predict the labels in multi-class multi labeling classification.

- **arXiv Papers**: We downloaded all of arXiv papers in quantitative finance[2] published between 2011 and 2018. Then we selected five major categories (subject tags): “q-fin.GN” (General Finance), “q-fin.PR” (Pricing of Securities), “q-fin.MF” (Mathematical Finance), “q-fin.ST” (Statistical Finance), and “q-fin.RM” (Risk Management). For pre-processing we removed the titles, author names and affiliations, abstracts, keywords and references. Then we tried to

[2]https://arXiv.org/archive/q-fin

Algorithm 2 Topological Feature Extraction from Word Embeddings

```
Input: word embedding representation of text:
    A matrix $\Psi_{T\times D}$ where $T$ is the number of tokens in the text and $D$ is the dimensionality of word embedding.
Output: embedding-based topological features of text:
    A vector of size $2 \times D$.
1:  for $d = 1$ to $D$ do
2:      Smooth d-th column of $\Psi$: Update the smoothed matrix $\tilde{\Psi}$ smoothing column $\Psi^{(d)}$ of $\Psi_{T\times D}$.
3:  end for
4:  for $i, j = 1$ to $D$ do
5:      Calculate distance between columns $\tilde{\Psi}^{(i)}$ and $\tilde{\Psi}^{(j)}$.
        $\theta(\tilde{\Psi}^{(i)}, \tilde{\Psi}^{(j)}) = \frac{1}{T}||\tilde{\Psi}^{(i)}|| \cdot ||\tilde{\Psi}^{(j)}||\{1 - cos(\tilde{\Psi}^{(i)}, \tilde{\Psi}^{(j)})\}$
6:  end for
7:  Apply persistent homology on $\Theta_{D\times D} = [\theta(\tilde{\Psi}^{(i)}, \tilde{\Psi}^{(j)})]$.
    Get persistence diagrams $PD_0(\Theta)$ and $PD_1(\Theta)$ for components and loops respectively.
8:  for $d = 1$ to $D$ do
9:      Make the persistence diagrams excluding $d$-th column and row from $\Theta$, i.e., $PD_0(\Theta \setminus d)$ and $PD_1(\Theta \setminus d)$.
10:     Calculate Wasserstein distance of persistence diagram including and excluding dimension $d$.
            $V_0(d) = Wasserstein\{PD_0(\Theta), PD_0(\Theta \setminus d)\}$
            $V_1(d) = Wasserstein\{PD_1(\Theta), PD_1(\Theta \setminus d)\}$
11: end for
12: return $\Omega_0$ and $\Omega_1$ as
            $\Omega_0 = [V_0(d)] \quad ; \quad d = 1, \ldots, D$
            $\Omega_1 = [V_1(d)] \quad ; \quad d = 1, \ldots, D$
```

predict the subjects solely based on the paper body.

- **IMDB Movie Review** [123]: Using IMDB reviews annotated by positive/negative label, we examined the topological algorithm on word embeddings for binary sentiment classification task.

Table 4.2 contains the specifications of both data sets. Note that each records in the arXiv data set may have more that one label. The histogram of number of labels for each record is shown in Figure 4.7. As shown in the histogram, the majority of records in arXiv data set are tagged with only a single label.

Table 4.2: Data Specification for arXiv papers and IMDB review data sets.

Specification	arXiv Quant. Fin. Papers	IMDB Movie Reviews
Labels	5 (Multi-label)	2
Clean Records	4601	6000
Length of Records	8456.9 ± 6395.8	540.5 ± 171.9
Frequency of Labels	$q\text{-}fin.GN$: 1258 $q\text{-}fin.ST$: 1144 $q\text{-}fin.MF$: 977 $q\text{-}fin.PR$: 907 $q\text{-}fin.RM$: 913	$Positive$: 3000 $Negative$: 3000

In practice, for many of the classification and clustering tasks in text processing, the data covers only very short documents (e.g., a limited data set of social media posts). Therefore a big challenge is training word embedding models on short documents. Such a challenge is beyond the scope of this study and we will use pre-trained versions of word embeddings that are previously trained on large corpora. Specifically, we use the following pre-trained models.

- GloVe [107] pre-trained on Wikipedia 2014 and Gigaword 5 with vocabulary

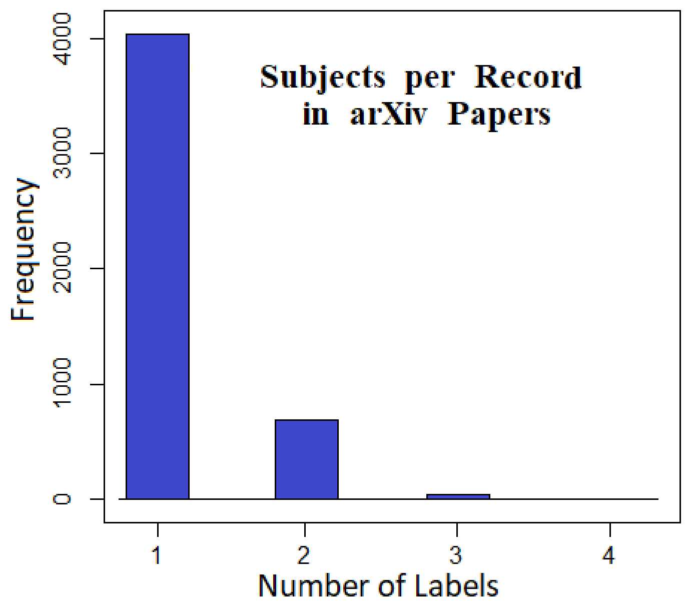


Figure 4.7: Histograms of number of labels per document in arXiv data set of papers.

size of 400K and 300d vectors[3].

- fastText [124, 125] pre-trained on Wikipedia 2017 with the vocabulary size of 1M and 300d vectors[4].
- ConceptNet Numberbatch [110] $v17.06$ with the vocabulary size of 400K and 300d vectors[5].

4.2.3 Results from TDA on Word Embeddings

We run our binary classification and multi-label multi-class classification on both data set using XGBoost [126, 127] with the parameters $eta = 0.25$, $max\_depth = 10$, $subsample = 0.5$, and $colsample\_bytree = 0.5$. In each data set 2/3 of the records were selected for train the model and the remaining 1/3 were used for testing. Table 4.4 and Table 4.5 show the results on arXiv paper data set and IMBD movie review

[3]http://nlp.stanford.edu/data/glove.6B.zip
[4]https://dl.fbaipublicfiles.com/fasttext/vectors-wiki/wiki.en.vec
[5]https://conceptnet.s3.amazonaws.com/downloads/2017/numberbatch/numberbatch-en-17.06.txt.gz

Table 4.3: Results per class on arXiv papers dataset using ConceptNet Numberbatch as pre-trained embedding and window size of 3.

Subject	Test Records	Precision	Recall	F1	Accuracy
q-fin.GN	410	73.2	68.5	0.708	83.8
q-fin.ST	396	70.2	67.5	0.688	83.6
q-fin.MF	306	66.0	45.6	0.539	77.5
q-fin.PR	305	69.5	55.2	0.615	82.7
q-fin.RM	307	62.5	61.0	0.617	84.5

data set, respectively. On each data set, we run the classifier using different pre-trained embedding models and different sliding window sizes to smooth the embedding signals. For both arXiv papers set and IMDB Movie Review data set, the best result is achieved using ConceptNet Numberbatch as pre-trained embedding and window size of 3. Detailed results for arXiv papers set are shown in Table 4.3.

To evaluate out results, for arXiv data set we run a convolutional neural network using the same pre-trained word embeddings. As shown in Table 4.4, our best configuration using our topological features outperforms the base CNN model in terms of accuracy and F1 score defined by Equation 4.9 and Equation 4.10 respectively.

$$\begin{aligned} Accuracy &= \frac{TP + TN}{P + N} \\ &= \frac{TP + TN}{TP + TN + FP + FN} \end{aligned} \tag{4.9}$$

$$F1\text{-}Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{4.10}$$

Here TP and TN are the numbers of true positives and true negatives and FP FN stand for the numbers of false positives and false negatives respectively. Precision and recall are defined in Equation 4.11 and Equation 4.12

Table 4.4: Results on arXiv papers dataset. The best result is achieved using ConceptNet Numberbatch as pre-trained embedding and window size of 3.

Model	Embedding	Window	Prec.	Rec.	F1	Acc.
Topology + XGBoost	fastText	3	61.9	55.4	0.575	80.1
Topology + XGBoost	GloVe	3	63.1	56.7	0.597	80.7
Topology + XGBoost	Numberbatch	3	**68.7**	**60.5**	**0.643**	**82.6**
Topology + XGBoost	fastText	5	60.8	54.7	0.576	79.8
Topology + XGBoost	GloVe	5	61.8	56.1	0.588	80.3
Topology + XGBoost	Numberbatch	5	65.5	58.4	0.617	81.6
Topology + XGBoost	fastText	7	58.9	54.4	0.566	79.5
Topology + XGBoost	GloVe	7	62.8	56.4	0.594	80.6
Topology + XGBoost	Numberbatch	7	65.7	57.7	0.614	81.3
Topology + XGBoost	fastText	7 expon.	60.3	54.6	0.573	79.7
Topology + XGBoost	GloVe	7 expon.	61.2	55.9	0.584	80.2
Topology + XGBoost	Numberbatch	7 expon.	66.4	59.6	0.628	82.2
CNN	fastText	-	57.1	64.3	0.605	80.0
CNN	GloVe	-	57.6	64.2	0.607	80.6
CNN	Numberbatch	-	**55.0**	**67.6**	**0.607**	**79.8**

$$Precision = \frac{TP}{FP + TP} \tag{4.11}$$

$$Recall = \frac{TP}{FN + TP} \tag{4.12}$$

For IMDB reviews data set, we compare our results to the previous results of Shauket et al. (2020) [128] lexicon based approach and Giatsoglou et al. (2017) [129] hybrid approach. The comparison reveals that our topological algorithm outperforms the previous models.

Table 4.5: Results on IMDB Movie Review dataset. The best result is achieved using ConceptNet Numberbatch as pre-trained embedding and window size of 3.

Model	Embedding	Window	Prec.	Rec.	F1	Acc.
Topology + XGBoost	fastText	3	84.8	85.8	0.853	85.4
Topology + XGBoost	GloVe	3	86.9	88.0	0.874	87.5
Topology + XGBoost	Numberbatch	3	**87.9**	**89.0**	**0.884**	**88.5**
Topology + XGBoost	fastText	5	84.2	85.2	0.847	84.8
Topology + XGBoost	GloVe	5	85.6	86.6	0.861	86.2
Topology + XGBoost	Numberbatch	5	86.5	87.6	0.870	87.1
Topology + XGBoost	fastText	7	82.8	83.8	0.833	83.4
Topology + XGBoost	GloVe	7	83.8	84.8	0.843	84.4
Topology + XGBoost	Numberbatch	7	85.3	86.3	0.858	85.9
Topology + XGBoost	fastText	7 expon.	84.3	85.3	0.848	84.9
Topology + XGBoost	GloVe	7 expon.	86.5	87.6	0.870	87.1
Topology + XGBoost	Numberbatch	7 expon.	87.0	88.1	0.875	87.6
Shauket et al. (2020) [128]	Lexicon based	-				86.7
Giatsoglou et al. (2017) [129]	Hybrid	-			**0.880**	**87.8**

4.3 Additive Information in TDA Features Using TF/IDF and Word Embeddings

As mentioned before, we may refine topological features out of TF/IDF space or word embedding representation of text. Previously we defined the method to extract the topological features from word embeddings space. Here we introduce a method that extracts topological features from TF/IDF space and then we add both set of features to a conventional text classifier and quantify the availability of additive information in the topological features as the change in the classification performance.

4.3.1 TDA on TF/IDF Vs. Word Embeddings

Our method of extracting embedding based topological features is described in Algorithm 2. As mentioned in Algorithm 2 working on textual documents represented in D-dimensional word embeddings, we get D features for topological dimension 0 (components) and another D features for topological dimension 1 (loops). We can use the resulted $2 \times D$ topological features to represent the text. To apply persistent homology on TF/IDF space, we follow the approach in [69], i.e., dividing the textual document to a fixed number of blocks and then searching for repetitive patterns in the text. In fact we analyze the topological properties of a graph where the vertices are coming from the TF/IDF vectors of the consequent blocks of the original text. Once again we utilize persistent homology to retrieve the geometric structures under different distance resolutions. The details of our method are described in Algorithm 3.

We divide each document into 10 consecutive blocks of equal size, we calculate TF/IDF vector for each block. We chose 10, but one may try different number of blocks for each document. However, we note that using a large number of blocks could make the TF/IDF vectors too sparse, so that comparing them would not be useful. For instance, if an average number of tokens in a document is only 200 tokens

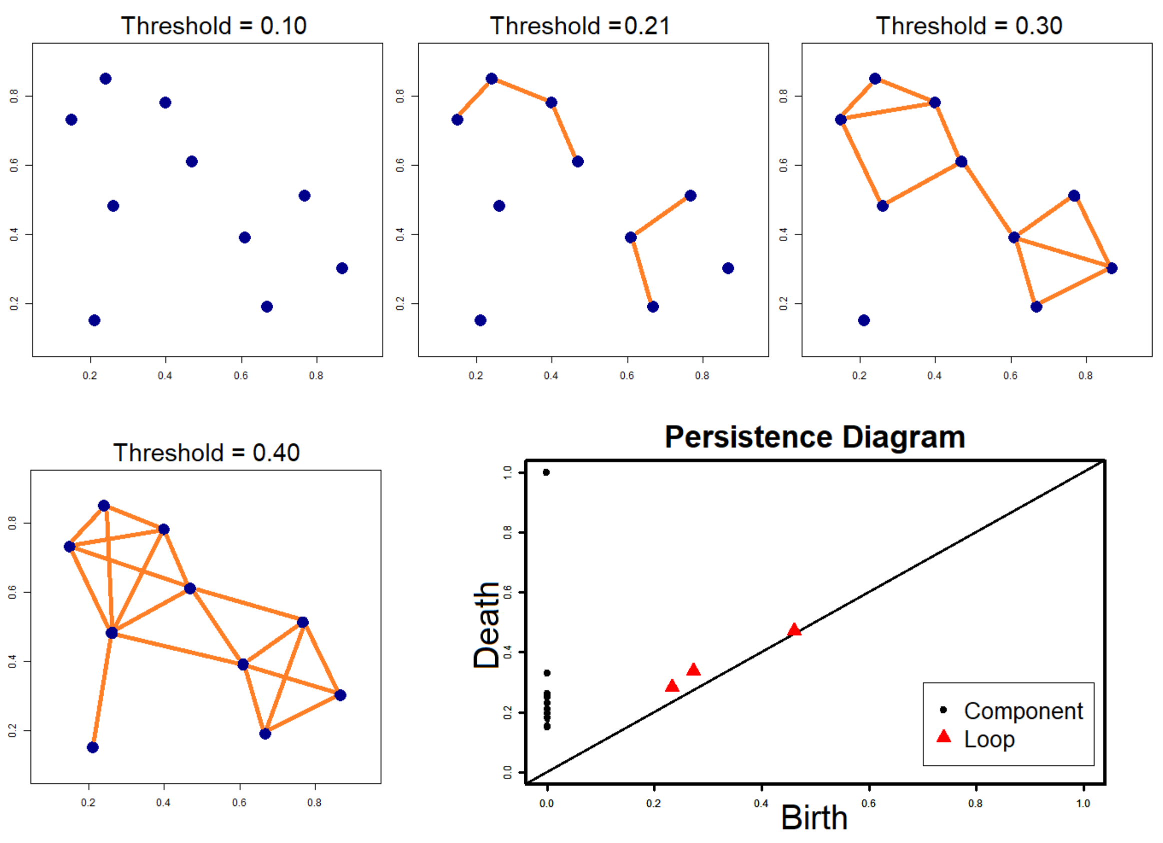


Figure 4.8: Working on the graph of 10 vertices, persistent homology thresholds the distance (e.g., cosine distance) among different nodes using all possible thresholds.The resulted edges for a few choices of threshold are shown here. Topological characteristics are summarized in the persistence diagram (bottom left).

and we divide each of the documents into 100 blocks, there would be two tokens in each block, and most of the blocks would have zero similarity.

In our experiments, we work on graphs of 10 vertices, where each vertex is represented by its TF/IDF vector. An example of such graphs is illustrated in Figure 4.8. The figure shows that when persistent homology is applied, the number of edges connecting the ten vertices will increase with the size of the radius (as we described in Chapter 2). The distance between two vertices is given by the cosine similarity of the vectors associated with each vertex.

With 10 vertices, in topological dimension 0 (components) we get exactly 9 diameters of birth and 9 diameters of death. Since for topological dimension 0 all of the birth diameters are always equal to zero, we only retrieve 9 death diameters.

For topological dimension 1 (loops), we may get different number of loops for different documents. Thus, if we retrieve all of birth and death diameters, we will get different numbers of features for different textual documents. Therefore, we summarize the information from topological dimension 1 (loops) in five statistically inferred features: number of loops, the average diameter of birth, the average diameter of duration, the standard deviation of birth diameters, and the standard deviation of duration diameters. This is similar to what Mittal and Gupta suggested in [28] to summarize persistence diagram— that is using six features from the persistence diagram including the number of holes, the average lifetime of holes, the maximum diameter of holes and the maximum distance between holes in each dimension. Here we utilize some similar features. The resulting 14 features (9 from dimension zero plus 5 from dimension one) represent patterns in the text. (As noted by [69] such representation may capture e.g. repetitive patterns of the text).

Among the five features we retrieve from TF/IDF representation in topological dimension 1, the number of loops is the simplest statistic we can define to quantify the shape of the graph of 10 text blocks (by counting them). The average diameter of birth represents the scale of the holes in the graph— that measures at which diameters the holes begin to appear on average. The standard deviation of birth diameter measures how diverse are such diameters. In other word, mean and standard deviation of birth diameters represent the *location* and *scale* [130] in the distribution of birth diameters, respectively. Working on duration diameters, once again we measure the location and scale of their distribution. Note that duration diameters represent the magnitude of loops in the graph of text blocks— they measure how persistent are the holes in the graph. We summarize the description of the five mentioned features in Table 4.6.

Algorithm 3 Topological Features from TF/IDF

```
Input: text:
    A array R of size T where T is number of tokens in text.
Output: TF/IDF based topological features of text:
    A vector of size 14.
1: Divide R to 10 equal-size arrays of size T/10.
            $$R = Concat(R^{(1)}, \dots, R^{(10)})$$
2: for $d = i$ to 10 do
3:     Calculate TF/IDF vector of $R^{(i)}$.
            $$\phi(i) = \text{TF/IDF}\{R^{(i)}\}$$
4: end for
5: Apply persistent homology on $\Phi = [\phi(i)]$ ; $i = 1, \dots, 10$.
6: Set $X = [x_1, \dots, x_9]$ where $x_i$'s are the diameters of deaths for components (dimension 0). We get exactly 9 death diameters.
7: For each loop (dimension 1), we have the diameters of birth and death.
   Calculate $Y = [y_1, \dots, y_5]$
        $y_1$ = number of loops
        $y_2$ = average diameter of birth
        $y_3$ = average diameter of duration
        $y_4$ = standard deviation of of birth diameters
        $y_5$ = standard deviation of duration diameters
   where duration diameter is defined as death diameter minus birth diameter.
8: return $X$ and $Y$
```

Table 4.6: Features from dimension 1 extracted from TF/IDF representation (the graph of 10 text blocks), with their other uses. Features 2,3, and 5, to our knowledge, were not previously used with TDA for classification, even though standard deviation is a most common statistic.

	Feature	**Loops Distribution Information**	**Prev. Applications**
1	Number of loops	Loops variety among text blocks	Used by Zhu [69] to find repetitive patterns in text.
2	Mean diameter of birth	Birth diameter location	
3	Std. Dev. of birth diameters	Birth diameter scale	
4	Mean duration (death - birth)	Duration location	Used by Mittal and Gupta [28] for dynamical system analysis.
5	Std. Dev. of duration	Duration scale	

4.3.2 Specification of the Experiment

We run both Algorithm 2 and Algorithm 3 on Wikipedia Movie Plots from Kaggle[6]. We selected the movie plots annotated by four major genres of Drama, Comedy, Action, and Romance. Keeping only the plots containing at least 200 words, we tried to predict the genres solely based on the plot texts.

The data set contains 11,500 total records. Each record may have been annotated by more than one labels. More specifications per class are shown in Table 4.2. We used 2/3 of the records for training and 1/3 for testing.

To represent the data in word embedding space, we used fastText [124, 125] pre-trained on Wikipedia 2017 with the vocabulary size of 1M and 300d vectors[7]. We chose fastText since in our initial experiment it showed slightly better performance compared to Google word2vec [108, 131, 132], GloVe [107], and Conceptnet numberbatch [110] pre-trained vectors. To apply persistent homology and extract topological

[6]https://www.kaggle.com/aminejallouli/genre-classification-based-on-wiki-movies-plots/data
[7]https://dl.fbaipublicfiles.com/fasttext/vectors-wiki/wiki.en.vec

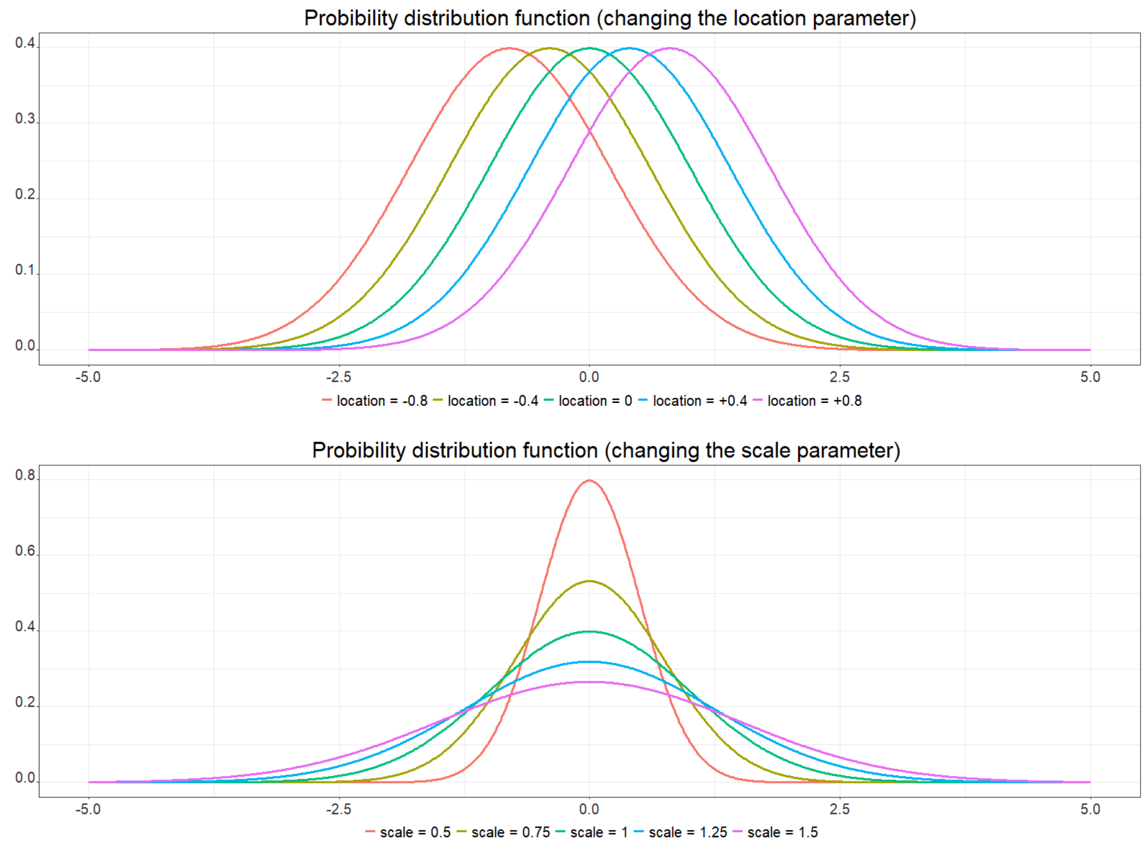


Figure 4.9: The effect of changing the location and the scale of a distribution. Changing the location will push the probability density function to left/right (top) while changing the scale will stretch/shrink the probability density function (bottom).

features we utilized Ripser [121] package. The TF/IDF vectors for Algorithm 3 were extracted with text2vec package [133].

4.3.3 Results of TDA using TF/IDF and Word Embeddings

For each record in the data set, we computed two sets of topological features based on word embeddings as in Algorithm 2 and TF/IDF space as in Algorithm 3. We will call these two sets of features *TP1* and *TP2*, respectively.

First, we fed $TP1$ to the XGBoost [127] classifier with $max\_depth = 2$, $eta = 1$, and 25 iterations. Then we tried adding $TP2$ features to the same classifier to boost the results. We also tried a Bidirectional LSTM to classify the records without using

Table 4.7: Number of records per class and overlaps among different classes.

Specification	Drama	Comedy	Action	Romance
Overlap with drama	-	524	223	379
Overlap with comedy	524	-	207	544
Overlap with action	223	207	-	117
Overlap with romance	379	544	117	-
Exclusive Records	4592	3302	1181	672
Total Records	5615	4477	1658	1614

our topological features. Our bidirectional LSTM model containing 64-dimensional main layer output was trained with a batch size of 32 in five epochs with adam optimizer [134].

While bidirectional LSTM showed stronger performance than the XGBoost model feeding our topological features, we assumed that there might be some exclusive information carried by our topological features that are not captured by the LSTM. Thus we tried combining the LSTM results with the XGBoost models. As one of the easiest ways to combine the results, we fed the probabilities (not the rounded predictions) returned by the two models (LSTM and XGBoost using $TP1$ and $TP2$) to a logistic regression model.

As shown in Table 4.8, our best ensemble model outperforms the LSTM accuracy and F1-score by 1.6% and 5.1%, respectively. The previous results[8] using linear Support Vector Classifier (SVC) and multinomial Naïve Bayes are also provided in the table. The detailed results per class are also provided in Table 4.9.

Note that the topological features that we extracted from the word embedding space (i.e., $TP1$) can classify the records alone with an accuracy comparable but not equal to the LSTM. On the other hand, the topological features extracted from TF/IDF space are primarily used to reflect some repetitive patterns in the text, as

[8]https://www.kaggle.com/aminejallouli/genre-classification-based-on-wiki-movies-plots/notebook

Table 4.8: Macro-average Results by different methods.

	Classifier	Pre.	Rec.	F1	Acc.
1	BiLSTM	68.0	59.7	0.608	76.2
2	XGBoost on TP1	59.6	53.2	0.560	71.1
3	XGBoost on TP1 & TP2	59.9	53.7	0.564	71.4
4	BiLSTM + XGBoost on TP1	67.8	64.8	0.656	77.3
5	BiLSTM + XGBoost on TP1 & TP2	68.5	64.6	0.659	77.8
	Previous Results (Linear SVC)				73.5
	Previous Results (Naïve Base)				73.3

Table 4.9: Accuracy per class using different methods. Here BiLSTM, XGB, XGB2, LR, and LR2 are the same as models 1 to 5 in Table 4.8. prev. SVC and prev. NB refer to the previous results using linear SVC and multinomial Naïve Bayes, respectively.

Class	BiLSTM	XGB	XGB2	LR	LR2	prev. SVC	prev. NB
action	87.7	86.7	86.9	89.3	88.9	81.5	82.7
comedy	75.6	69.0	69.1	76.9	77.7	74.6	73.3
drama	69.9	63.9	64.3	71.0	71.6	66.1	67.4
romance	87.6	86.0	85.9	87.8	87.8	88.3	84.3
macro-avg	76.2	71.1	71.4	77.3	77.8	73.5	73.3

Zhu [69] suggested in a similar study. However, as shown in Table 4.8 and Table 4.9, using the topological feature sets can boost the accuracy of classification in the ensemble model.

4.4 Discussion

In the proposed framework of extracting topological features without using conventional text representations (as introduced in Section 4.1), the results reveal that some homological fingerprints of the authors exist in their works. As the main contribution, we analyzed the special names (the leading characters through the books), made the graphs of proximity among the leading characters, and finally used those graphs to extract the homological features using persistent homology. One of the main properties of the framework we introduced is the robustness toward the translation and insensitivity to the documents size. Although the performance is not much reliable to apply it to many applied problems, we can still modify or generalize the framework to make it more accurate. For instance trying different model parameters, introducing different quantifiers and measures of distances on the topological graph, or even utilization of POS tags, location tags, or topics instead of the spacial names as we did, may lead to a better algorithm.

However, the method of extracting homological features without utilising conventional features is not easily applicable to the general problem of text classification. Therefore, we explored the possibility of extracting some topological features from widely used numerical text representations such as TF/IDF and word embeddings, as described in Section 4.2 and Section 4.3.

In Section 4.2, we introduced a novel method to define and extract topological features from word embedding representations of corpus and used them for text classification. We utilized persistent homology, the most common tool from TDA to interpret the embedding space of each textual documents. In our experiments, we showed that working on textual documents, our defined topological features can outperform conventional text analysis features. Specially when the textual documents are long, using these topological features can improve the results. However, in topological algorithm over the embeddings space, we are analyzing different embedding dimensions as time series. Thus, to achieve reasonable results, a large number of tokens in each textual document is required. We acknowledge this issue as the main limitation of our algorithm. Also, it is not easy to measure and/or interpret the impact of different parts of the text input on the output in the topological algorithm. This is one of the possible future directions for this study.

In Section 4.3, we described a framework to use holomogical features in addition to the conventional features for text classification. Here the focus is on introducing and examining two methods to apply TDA to text classification. Term frequency (or TF/IDF) and word embeddings are the most frequently used methods to translate the text into numerical data. Therefore they deserve to be examined, as a priority, for potential to reveal their hidden dimensions by applying topological methods. First, we introduce a novel method of using word embeddings where we view text documents as time series. We believe this method shows great promise, since it can be applied to documents irrespective of their length (with some likely limitations), and it encodes the temporal succession words in a latent semantic space. Our algorithm analyzes the topology of the embedding space to discover relations among different embedding dimensions of the analyzed text. The precise nature of this space is not clear to us at this point. However, we know it is there, because our experiments show its influence on the accuracy of classification.

In the second experiment, working with TF/IDF representations of textual documents, we use a method that divides the text to a fixed number of blocks, analyzes topological structure of the relations among different blocks and summarizes the results. As with the first method, this topological summary consists of numerical features derived from the persistence diagram of each document. And as in first case, it improves the accuracy of classification, proving the existence of the latent topological dimension (speaking metaphorically).

The intuitive idea behind both experiments has to do with the central premise of topological data analysis, namely that when examining a cloud of data points at different resolutions, the emerging diagrams encode global geometric properties of the point as shown in Figure 2.4 and later in Figure 4.8. There we observe, with the change of the threshold, i.e. the distance at which we add connections to the points, new elements are added to the persistence diagram, culminating in a clear circle, or torus-like signature in in Figure 2.4 (shown as the long line in the right panel), and a more complex representation of the geometry in Figure 4.8. In our case we measure, and use as features for machine learning the birth and death diameters in dimension 0 and 1, as well as their derivatives that is the number of holes, the average divided by the standard deviation of death diameter, and the same ratio for the duration (death - birth).

We used two different methods to extract topological features from text and applied them to the task of document classification. The first method converts text, represented as a sequence of word embeddings into a high dimensional time series, which at the end is analyzed using the machinery of topological data analysis, namely homological persistence. The second method augments the classical TF/IDF representation of the text with topological features.

Specifically, we have leveraged existing word embeddings along with topology of text to show that such structure can carry some useful information for machine learning classifiers to learn from.

As we have shown in the results, while a classifier utilizing only topological features may fail to outperform more conventional models like bidirectional LSTMs, these topological features are capable of carrying some exclusive information that is not captured by the conventional text analysis methods. Therefore, adding these features to more conventional features models can boost the results. In our experiment, adding using topological features in the ensemble model resulted in 4.9% increase in recall, 0.5% increase in precision, and 5.1% increase in F1 score.

Briefly, our contributions are as follows:

- We introduced a new algorithm of extracting topological features from text, namely by converting a sequence of word embeddings into a time series, and analyzing the dimensions of the resulting series for topological persistence.
- This algorithm works with documents of any length and, importantly, preserves the word order in its representation.
- We have shown that this new method produces features of value for the task of document classification.
- We showed that even if the representation of documents is derived from the standard TF/IDF matrix, similarly produced topological features improve the accuracy of classification.

We end with a discussion, including some of the limitations, and open problems. The strength of our algorithm for analyzing documents as a time series of embeddings is in its universal applicability, irrespective of the length of the document. The second important property is using the word order. Finally, the algorithm produces the representation in one pass.

However, one of the limitations of our methodology is the size of block of text. Regarding the embedding based topological features, the topological structure of a short text would not be stable. Also due to lack of context, the embedding may not be able to provide enough information for classification tasks. Similarly, using its

TF/IDF vectors on short documents, can result in poor simplicial shapes, when we divide our text in blocks of 10, as in Section 4.3. That is, a set of separate dots in the space most of which are not connected at all. In such a case, it is challenging to find informative topological structure in text.

Proving the value of the methods used in this dissertation for other natural language processing tasks, such as summarization, entity extraction or question answering, is both a limitation of this work, and an open problem.

CHAPTER 5: Summary and Future Direction

While the feasibility of topological data analysis has been explored by many studies on numeric data sets, it is still a challenging task to apply it to text. As the primary goal in topological data analysis is defining and then quantifying the underlying shapes in (numeric) data, defining shapes in textual documents is much more challenging, even though the geometries of the vector spaces and the conceptual spaces are clearly relevant for information retrieval and semantics.

We examined three different methods to define and extract topological features from textual documents, using as the underlying representations of words the two most popular methods, namely term frequency vectors and word embeddings, and also without using any conventional features. Here are the short descriptions of the methods:

1. To extract topological features without using conventional features, we analyzed the graph of appearance/co-appearance of the leading characters through long documents. Choosing some publicly available novels as the symbolic instances of long textual documents, tries to predict the authors based on the topological properties of such graphs.

2. To extract topological features from the embedding space, we interpreted the word embedding representation of the text as a high dimensional time series, and then we analyzed the topology of the underlying graph whose vertices correspond to the different embedding dimensions.

3. For topological data analysis on the term frequency space, we analyzed the topology of the graph where the vertices represent the TF/IDF vectors of different blocks in the text.

In all cases, we applied persistent homology to retrieve the geometric structures of the text under different distance resolutions. Our results show that the topological features are capturing some exclusive information that is not carried by the conventional text representations.

The idea of extracting topological features without using conventional features was to investigate our hypothesis that (a) different novels by the same author have similar persistence diagrams and (b) different authors usually have significantly different persistence diagrams. The average accuracy of 77% for the author prediction task suggests that there are topological signatures (topological styles) in the novels that identify the authors.

Feeding topological features refined from word embeddings space, we achieved the F1 score of 64.3% on arXiv papers classification that beats our convolutional neural network model by 3.6%. Using the same methodology on IMDB Movie Review data set, we reached to the F1 score of 88.4% which is 0.4% above the previous results using lexicon based and hybrid approaches.

In our experiment of using topological features for additive information, using topological features (extracted from TF/IDF and word embeddings spaces) in the ensemble model resulted in 4.9% increase in recall, 0.5% increase in precision, and 5.1% increase in F1 score, on the Wikipedia movie plot data set.

Based on the above, we have shown that topological methods deserve deeper examination as a tool for text analysis. We believe that as with the geometries of vector spaces and conceptual spaces mentioned earlier in Section 4.2 and Section 4.3, the topological features, which capture certain geometric invariants, are relevant for text analytics and semantics of natural language.

In our future work, on extraction of topological features without using conventional features, we will analyze the co-appearances of more entity types. Currently, we have analyzed only the name entities tagged as ‘person’. However, the co-appearances of locations, different POS, etc. might be informative as well. We validated our method on a set of novels, for which the name entities are intuitively among the most important tokens. But for different document types, that is not necessarily true. That is why extending our method to analyze other entity types is worth of investigating.

Here we mainly focused on text classification. We will extend and apply our methods for other natural language processing tasks, such as summarization, entity extraction or question answering.

We will investigate the possibility of connecting our work on topology of text with the work on the understanding of topological properties of deep neural networks, exemplified e.g. by [135] and [91].

We tried extracting topological features without using conventional features, using TF/IDF space, and using word embeddings space. Another interesting input for the topological data analysis would be the attention models— that is the attention matrix.

In our method of extracting topological features from TF/IDF space we were dividing each document to a fixed number of blocks, e.g. 10 consecutive blocks. We know that persistent homology utilises a filtration on distances, i.e., defining the ϵ-distances around the data points and then increasing ϵ from 0 to ∞. We may investigate the performance of a 2D filtration where not only we try any possible ϵ in $(0, \infty)$, but we also try all possible numbers of blocks in $\{2, \ldots, L\}$ where L is the size of the textual document.

A technical open problem is to find the actual text behind the topological structures. This is a challenge to be addressed in our future work.

REFERENCES

[1] F. Aurenhammer, "Voronoi diagrams-a survey of a fundamental geometric data structure," *ACM Computing Surveys (CSUR)*, vol. 23, no. 3, pp. 345–405, 1991.

[2] A. Zomorodian, "Topological data analysis," *Advances in applied and computational topology*, vol. 70, pp. 1–39, 2012.

[3] G. Carlsson, "Topology and data," *Bulletin of the American Mathematical Society*, vol. 46, no. 2, pp. 255–308, 2009.

[4] H. Edelsbrunner and J. Harer, "Persistent homology-a survey," *Contemporary mathematics*, vol. 453, pp. 257–282, 2008.

[5] R. Ghrist, "Barcodes: the persistent topology of data," *Bulletin of the American Mathematical Society*, vol. 45, no. 1, pp. 61–75, 2008.

[6] L. M. Chen, Z. Su, and B. Jiang, *Mathematical Problems in Data Science*. Springer, 2015.

[7] A. Zomorodian, "Computational topology," in *Algorithms and theory of computation handbook*, pp. 3.3–3.4, Chapman & Hall/CRC, 2010.

[8] H. Bass, "Euler characteristics and characters of discrete groups," *Inventiones mathematicae*, vol. 35, no. 1, pp. 155–196, 1976.

[9] H. Edelsbrunner, D. Letscher, and A. Zomorodian, "Topological persistence and simplification," in *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pp. 454–463, IEEE, 2000.

[10] A. Zomorodian and G. Carlsson, "Computing persistent homology," *Discrete & Computational Geometry*, vol. 33, no. 2, pp. 249–274, 2005.

[11] E. Munch, "A user's guide to topological data analysis," *Journal of Learning Analytics*, vol. 4, no. 2, pp. 47–61, 2017.

[12] A. Collins, A. Zomorodian, G. Carlsson, and L. J. Guibas, "A barcode shape descriptor for curve point cloud data," *Computers & Graphics*, vol. 28, no. 6, pp. 881–894, 2004.

[13] G. Petri, M. Scolamiero, I. Donato, and F. Vaccarino, "Topological strata of weighted complex networks," *PloS one*, vol. 8, no. 6, p. e66506, 2013.

[14] P. Bubenik, "Statistical topological data analysis using persistence landscapes," *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 77–102, 2015.

[15] S. C. Johnson, “Hierarchical clustering schemes,” *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.

[16] C. M. Pereira and R. F. de Mello, “Persistent homology for time series and spatial data clustering,” *Expert Systems with Applications*, vol. 42, no. 15-16, pp. 6026–6038, 2015.

[17] F. A. Khasawneh and E. Munch, “Stability determination in turning using persistent homology and time series analysis,” in *ASME 2014 International Mechanical Engineering Congress and Exposition*, pp. V04BT04A038–V04BT04A038, American Society of Mechanical Engineers, 2014.

[18] J. A. Perea and J. Harer, “Sliding windows and persistence: An application of topological methods to signal analysis,” *Foundations of Computational Mathematics*, vol. 15, no. 3, pp. 799–838, 2015.

[19] S. Maletić, Y. Zhao, and M. Rajković, “Persistent topological features of dynamical systems,” *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 26, no. 5, p. 053105, 2016.

[20] B. J. Stolz, H. A. Harrington, and M. A. Porter, “Persistent homology of time-dependent functional networks constructed from coupled time series,” *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 27, no. 4, p. 047410, 2017.

[21] P. Skraba, V. de Silva, and M. Vejdemo-Johansson, “Topological analysis of recurrent systems,” in *NIPS 2012 Workshop on Algebraic Topology and Machine Learning, December 8th, Lake Tahoe, Nevada*, pp. 1–5, 2012.

[22] N. H. Packard, J. P. Crutchfield, J. D. Farmer, and R. S. Shaw, “Geometry from a time series,” *Physical review letters*, vol. 45, no. 9, p. 712, 1980.

[23] F. Takens, “Detecting strange attractors in turbulence,” in *Dynamical systems and turbulence, Warwick 1980*, pp. 366–381, Springer, 1981.

[24] J. Berwald and M. Gidea, “Critical transitions in a model of a genetic regulatory system,” *arXiv preprint arXiv:1309.7919*, 2013.

[25] J. Berwald, M. Gidea, and M. Vejdemo-Johansson, “Automatic recognition and tagging of topologically different regimes in dynamical systems,” *arXiv preprint arXiv:1312.2482*, 2013.

[26] J. Garland, E. Bradley, and J. D. Meiss, “Exploring the topology of dynamical reconstructions,” *Physica D: Nonlinear Phenomena*, vol. 334, pp. 49–59, 2016.

[27] V. De Silva and G. E. Carlsson, “Topological estimation using witness complexes.,” *SPBG*, vol. 4, pp. 157–166, 2004.

[28] K. Mittal and S. Gupta, "Topological characterization and early detection of bifurcations and chaos in complex systems using persistent homology," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 27, no. 5, p. 051102, 2017.

[29] F. Chazal, L. J. Guibas, S. Y. Oudot, and P. Skraba, "Persistence-based clustering in riemannian manifolds," *Journal of the ACM (JACM)*, vol. 60, no. 6, p. 41, 2013.

[30] H.-W. Chang, S. Bacallado, V. S. Pande, and G. E. Carlsson, "Persistent topology and metastable state in conformational dynamics," *PloS one*, vol. 8, no. 4, p. e58699, 2013.

[31] F. A. Khasawneh and E. Munch, "Exploring equilibria in stochastic delay differential equations using persistent homology," in *ASME 2014 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, American Society of Mechanical Engineers Digital Collection, 2014.

[32] F. A. Khasawneh and E. Munch, "Chatter detection in turning using persistent homology," *Mechanical Systems and Signal Processing*, vol. 70, pp. 527–541, 2016.

[33] F. A. Khasawneh, E. Munch, and J. A. Perea, "Chatter classification in turning using machine learning and topological data analysis," *IFAC-PapersOnLine*, vol. 51, no. 14, pp. 195–200, 2018.

[34] G. Quintana and J. Ciurana, "Chatter in machining processes: A review," *International Journal of Machine Tools and Manufacture*, vol. 51, no. 5, pp. 363–376, 2011.

[35] J. A. Perea, "Persistent homology of toroidal sliding window embeddings," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pp. 6435–6439, IEEE, 2016.

[36] M. Gidea, "Topological data analysis of critical transitions in financial networks," in *International Conference and School on Network Science*, pp. 47–59, Springer, 2017.

[37] H. Edelsbrunner and J. Harer, *Computational topology: an introduction.* American Mathematical Soc., 2010.

[38] A. Nobi, S. Lee, D. H. Kim, and J. W. Lee, "Correlation and network topologies in global and local stock indices," *Physics Letters A*, vol. 378, no. 34, pp. 2482–2489, 2014.

[39] M. Gidea and Y. Katz, "Topological data analysis of financial time series: Landscapes of crashes," *Physica A: Statistical Mechanics and its Applications*, vol. 491, pp. 820–834, 2018.

[40] N. Sanderson, E. Shugerman, S. Molnar, J. D. Meiss, and E. Bradley, "Computational topology techniques for characterizing time-series data," in *International Symposium on Intelligent Data Analysis*, pp. 284–296, Springer, 2017.

[41] V. Robins and K. Turner, "Principal component analysis of persistent homology rank functions with case studies of spatial point patterns, sphere packing and colloids," *Physica D: Nonlinear Phenomena*, vol. 334, pp. 99–117, 2016.

[42] A. Myers, E. Munch, and F. A. Khasawneh, "Persistent homology of complex networks for dynamic state detection," *Physical Review E*, vol. 100, no. 2, p. 022314, 2019.

[43] S. Tymochko, E. Munch, J. Dunion, K. Corbosiero, and R. Torn, "Using persistent homology to quantify a diurnal cycle in hurricanes," *Pattern Recognition Letters*, 2020.

[44] S. Emrani, T. Gentimis, and H. Krim, "Persistent homology of delay embeddings and its application to wheeze detection," *IEEE Signal Processing Letters*, vol. 21, no. 4, pp. 459–463, 2014.

[45] F. A. Khasawneh and E. Munch, "Topological data analysis for true step detection in periodic piecewise constant signals," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 474, no. 2218, p. 20180027, 2018.

[46] C. M. Topaz, L. Ziegelmeier, and T. Halverson, "Topological data analysis of biological aggregation models," *PloS one*, vol. 10, no. 5, p. e0126383, 2015.

[47] V. Venkataraman, K. N. Ramamurthy, and P. Turaga, "Persistent homology of attractors for action recognition," in *Image Processing (ICIP), 2016 IEEE International Conference on*, pp. 4150–4154, IEEE, 2016.

[48] J. A. Perea, A. Deckard, S. B. Haase, and J. Harer, "Sw1pers: Sliding windows and 1-persistence scoring; discovering periodicity in gene expression time series data," *BMC bioinformatics*, vol. 16, no. 1, p. 257, 2015.

[49] L. M. Seversky, S. Davis, and M. Berger, "On time-series topological data analysis: New data and opportunities," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2016 IEEE Conference on*, pp. 1014–1022, IEEE, 2016.

[50] S. A. Tabrizi, A. Shakery, H. Zamani, and M. A. Tavallaei, "Person: Personalized information retrieval evaluation based on citation networks," *Information Processing & Management*, vol. 54, no. 4, pp. 630–656, 2018.

[51] H. F. de Arruda, V. Q. Marinho, L. d. F. Costa, and D. R. Amancio, "Paragraph-based representation of texts: A complex networks approach," *Information Processing & Management*, vol. 56, no. 3, pp. 479–494, 2019.

[52] L. Song, Y. Zhang, Z. Wang, and D. Gildea, "A graph-to-sequence model for amr-to-text generation," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1616–1626, 2018.

[53] W. Li, R. Peng, Y. Wang, and Z. Yan, "Knowledge graph based natural language generation with adapted pointer-generator networks," *Neurocomputing*, vol. 382, pp. 174–187, 2020.

[54] M. Franco-Salvador, P. Rosso, and M. Montes-y Gómez, "A systematic study of knowledge graph analysis for cross-language plagiarism detection," *Information Processing & Management*, vol. 52, no. 4, pp. 550–570, 2016.

[55] Y. Deng, D. Lu, D. Huang, C.-J. Chung, and F. Lin, "Knowledge graph based learning guidance for cybersecurity hands-on labs," in *Proceedings of the ACM Conference on Global Computing Education*, pp. 194–200, 2019.

[56] R. Wang, M. Wang, J. Liu, M. Cochez, and S. Decker, "Structured query construction via knowledge graph embedding," *Knowledge and Information Systems*, pp. 1–28, 2019.

[57] Z. Wang, S. Gu, X. Zhao, and X. Xu, "Graph-based review spammer group detection," *Knowledge and Information Systems*, vol. 55, no. 3, pp. 571–597, 2018.

[58] S. Noekhah, N. binti Salim, and N. H. Zakaria, "Opinion spam detection: Using multi-iterative graph-based model," *Information Processing & Management*, vol. 57, no. 1, p. 102140, 2020.

[59] I. Ali and A. Melton, "Graph-based semantic learning, representation and growth from text: A systematic review," in *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*, pp. 118–123, IEEE, 2019.

[60] W. Etaiwi and A. Awajan, "Graph-based arabic text semantic representation," *Information Processing & Management*, vol. 57, no. 3, p. 102183, 2020.

[61] Z. Yang, W. Cohen, and R. Salakhudinov, "Revisiting semi-supervised learning with graph embeddings," in *International Conference on Machine Learning*, pp. 40–48, 2016.

[62] H. Van Lierde and T. W. Chow, "Query-oriented text summarization based on hypergraph transversals," *Information Processing & Management*, vol. 56, no. 4, pp. 1317–1338, 2019.

[63] A. Jain, S. Vij, and D. K. Tayal, “Text summarization using wordnet graph based sentence ranking,” in *Proceedings of 2nd International Conference on Communication, Computing and Networking*, pp. 711–715, Springer, 2019.

[64] C. Hark and A. Karcı, “Karcı summarization: A simple and effective approach for automatic text summarization using karcı entropy,” *Information Processing & Management*, vol. 57, no. 3, p. 102187, 2020.

[65] G. Ercan and F. Haziyev, “Synset expansion on translation graph for automatic wordnet construction,” *Information Processing & Management*, vol. 56, no. 1, pp. 130–150, 2019.

[66] M. Chen, Y. Tian, M. Yang, and C. Zaniolo, “Multilingual knowledge graph embeddings for cross-lingual knowledge alignment,” in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pp. 1511–1517, 2017.

[67] X. Chen, S. Jia, and Y. Xiang, “A review: Knowledge reasoning over knowledge graph,” *Expert Systems with Applications*, p. 112948, 2019.

[68] H. Wagner, P. Dłotko, and M. Mrozek, “Computational topology in text mining,” in *Computational Topology in Image Context*, pp. 68–78, Springer, 2012.

[69] X. Zhu, “Persistent homology: An introduction and a new text representation for natural language processing.,” in *IJCAI*, pp. 1953–1959, 2013.

[70] P. Doshi and W. Zadrozny, “Movie genre detection using topological data analysis,” in *International Conference on Statistical Language and Speech Processing*, pp. 117–128, Springer, 2018.

[71] H. Guan, W. Tang, H. Krim, J. Keiser, A. Rindos, and R. Sazdanovic, “A topological collapse for document summarization,” in *Signal Processing Advances in Wireless Communications (SPAWC), 2016 IEEE 17th International Workshop on*, pp. 1–5, IEEE, 2016.

[72] A. C. Wilkerson, T. J. Moore, A. Swami, and H. Krim, “Simplifying the homology of networks via strong collapses,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 5258–5262, IEEE, 2013.

[73] K. Almgren, M. Kim, and J. Lee, “Mining social media data using topological data analysis,” in *Information Reuse and Integration (IRI), 2017 IEEE International Conference on*, pp. 144–153, IEEE, 2017.

[74] G. Singh, F. Mémoli, and G. E. Carlsson, “Topological methods for the analysis of high dimensional data sets and 3d object recognition.,” in *SPBG*, pp. 91–100, 2007.

[75] K. Almgren, M. Kim, and J. Lee, "Extracting knowledge from the geometric shape of social network data using topological data analysis," *Entropy*, vol. 19, no. 7, p. 360, 2017.

[76] N. H. Christianson, A. S. Blevins, and D. S. Bassett, "Architecture and evolution of semantic networks in mathematics texts," *arXiv preprint arXiv:1908.04911*, 2019.

[77] P. Torres-Tramón, H. Hromic, and B. R. Heravi, "Topic detection in twitter using topology data analysis," in *International Conference on Web Engineering*, pp. 186–197, Springer, 2015.

[78] I.-J. Chiang, "Discover the semantic topology in high-dimensional data," *Expert Systems with Applications*, vol. 33, no. 1, pp. 256–262, 2007.

[79] W. Zadrozny and L. Garbayo, "A sheaf model of contradictions and disagreements. preliminary report and discussion," *arXiv preprint arXiv:1801.09036*, 2018.

[80] K. Savle, W. Zadrozny, and M. Lee, "Topological data analysis for discourse semantics?," in *Proceedings of the 13th International Conference on Computational Semantics-Student Papers*, pp. 34–43, 2019.

[81] G. Carlsson, T. Ishkhanov, V. De Silva, and A. Zomorodian, "On the local behavior of spaces of natural images," *International journal of computer vision*, vol. 76, no. 1, pp. 1–12, 2008.

[82] H. Adams and G. Carlsson, "On the nonlinear statistics of range image patches," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 110–117, 2009.

[83] J. A. Perea and G. Carlsson, "A klein-bottle-based dictionary for texture representation," *International journal of computer vision*, vol. 107, no. 1, pp. 75–97, 2014.

[84] V. De Silva and R. Ghrist, "Coordinate-free coverage in sensor networks with controlled boundaries via homology," *The International Journal of Robotics Research*, vol. 25, no. 12, pp. 1205–1222, 2006.

[85] V. De Silva, R. Ghrist, *et al.*, "Coverage in sensor networks via persistent homology," *Algebraic & Geometric Topology*, vol. 7, no. 1, pp. 339–358, 2007.

[86] E. Munch, M. Shapiro, and J. Harer, "Failure filtrations for fenced sensor networks," *The International Journal of Robotics Research*, vol. 31, no. 9, pp. 1044–1056, 2012.

[87] H. Adams and G. Carlsson, "Evasion paths in mobile sensor networks," *The International Journal of Robotics Research*, vol. 34, no. 1, pp. 90–104, 2015.

[88] R. Ghrist and S. Krishnan, "Positive alexander duality for pursuit and evasion," *SIAM Journal on Applied Algebra and Geometry*, vol. 1, 2017.

[89] J. Reininghaus, S. Huber, U. Bauer, and R. Kwitt, "A stable multi-scale kernel for topological machine learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4741–4748, 2015.

[90] C. Hofer, R. Kwitt, M. Niethammer, and A. Uhl, "Deep learning with topological signatures," in *Advances in Neural Information Processing Systems*, pp. 1633–1643, 2017.

[91] W. H. Guss and R. Salakhutdinov, "On characterizing the capacity of neural networks using algebraic topology," *arXiv preprint arXiv:1802.04443*, 2018.

[92] D. Pachauri, C. Hinrichs, M. K. Chung, S. C. Johnson, and V. Singh, "Topology-based kernels with application to inference problems in alzheimer's disease," *IEEE transactions on medical imaging*, vol. 30, no. 10, pp. 1760–1770, 2011.

[93] R. Eldan and O. Shamir, "The power of depth for feedforward neural networks," in *Conference on Learning Theory*, pp. 907–940, 2016.

[94] W. H. Guss, "Deep function machines: Generalized neural networks for topological layer expression," *arXiv preprint arXiv:1612.04799*, 2016.

[95] B. Benatallah, S. Venugopal, S. H. Ryu, H. R. Motahari-Nezhad, W. Wang, *et al.*, "A systematic review and comparative analysis of cross-document coreference resolution methods and tools," *Computing*, vol. 99, no. 4, pp. 313–349, 2017.

[96] W. Zhang, T. Yoshida, and X. Tang, "A comparative study of tf* idf, lsi and multi-words for text classification," *Expert Systems with Applications*, vol. 38, no. 3, pp. 2758–2765, 2011.

[97] A. Mishra and S. Vishwakarma, "Analysis of tf-idf model and its variant for document retrieval," in *2015 international conference on computational intelligence and communication networks (cicn)*, pp. 772–776, IEEE, 2015.

[98] J. Ramos *et al.*, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the first instructional conference on machine learning*, vol. 242, pp. 133–142, Piscataway, NJ, 2003.

[99] A. Aizawa, "An information-theoretic perspective of tf–idf measures," *Information Processing & Management*, vol. 39, no. 1, pp. 45–65, 2003.

[100] H. Peng, J. Li, Y. He, Y. Liu, M. Bao, L. Wang, Y. Song, and Q. Yang, "Large-scale hierarchical text classification with recursively regularized deep graph-cnn," in *Proceedings of the 2018 World Wide Web Conference*, pp. 1063–1072, 2018.

[101] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for text classification," *arXiv preprint arXiv:1606.01781*, 2016.

[102] J. Liu, W.-C. Chang, Y. Wu, and Y. Yang, "Deep learning for extreme multi-label text classification," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 115–124, 2017.

[103] P. Wang, Y. Qian, F. K. Soong, L. He, and H. Zhao, "Word embedding for recurrent neural network based tts synthesis," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4879–4883, IEEE, 2015.

[104] H. Park, S. Cho, and J. Park, "Word rnn as a baseline for sentence completion," in *2018 IEEE 5th International Congress on Information Science and Technology (CiSt)*, pp. 183–187, IEEE, 2018.

[105] A. Seyeditabari and W. Zadrozny, "Can word embeddings help find latent emotions in text? preliminary results," in *The Thirtieth International Flairs Conference*, 2017.

[106] A. N. Jagannatha and H. Yu, "Bidirectional rnn for medical event detection in electronic health records," in *Proceedings of the conference. Association for Computational Linguistics. North American Chapter. Meeting*, vol. 2016, p. 473, NIH Public Access, 2016.

[107] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[108] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[109] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[110] R. Speer, J. Chin, and C. Havasi, "Conceptnet 5.5: An open multilingual graph of general knowledge," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[111] S. Gholizadeh, A. Seyeditabari, and W. Zadrozny, “Topological signature of 19th century novelists: Persistent homology in text mining,” *Big Data and Cognitive Computing*, vol. 2, no. 4, p. 33, 2018.

[112] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, “The stanford corenlp natural language processing toolkit,” in *Association for Computational Linguistics (ACL) System Demonstrations*, pp. 55–60, 2014.

[113] S. Vallender, “Calculation of the wasserstein distance between probability distributions on the line,” *Theory of Probability & Its Applications*, vol. 18, no. 4, pp. 784–786, 1974.

[114] D. R. Sheehy, “Linear-size approximations to the vietoris–rips filtration,” *Discrete & Computational Geometry*, vol. 49, no. 4, pp. 778–796, 2013.

[115] K. Turner, “Rips filtrations for quasimetric spaces and asymmetric functions with stability results,” *Algebraic & Geometric Topology*, vol. 19, no. 3, pp. 1135–1170, 2019.

[116] T. K. Dey, D. Shi, and Y. Wang, “Simba: An efficient tool for approximating rips-filtration persistence via sim plicial ba tch collapse,” *Journal of Experimental Algorithmics (JEA)*, vol. 24, no. 1, pp. 1–16, 2019.

[117] B. T. Fasy, J. Kim, F. Lecci, and C. Maria, “Introduction to the r package tda,” *arXiv preprint arXiv:1411.1830*, 2014.

[118] J. Allaire, “Rstudio: integrated development environment for r,” *Boston, MA*, vol. 537, p. 538, 2012.

[119] RStudio Team, *RStudio: Integrated Development Environment for R*. RStudio, Inc., Boston, MA, 2015.

[120] K. Clark and C. D. Manning, “Entity-centric coreference resolution with model stacking,” in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1405–1415, 2015.

[121] U. Bauer, “Ripser: efficient computation of vietoris-rips persistence barcodes,” *arXiv preprint arXiv:1908.02518*, 2019.

[122] U. Bauer, “Ripser: a lean c++ code for the computation of vietoris–rips persistence barcodes,” *Software available at https://github. com/Ripser/ripser*, 2017.

[123] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pp. 142–150, Association for Computational Linguistics, 2011.

[124] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv preprint arXiv:1607.04606*, 2016.

[125] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," *arXiv preprint arXiv:1607.01759*, 2016.

[126] T. Chen, T. He, M. Benesty, V. Khotilovich, and Y. Tang, "Xgboost: extreme gradient boosting," *R package version 0.4-2*, pp. 1–4, 2015.

[127] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, ACM, 2016.

[128] Z. Shaukat, A. A. Zulfiqar, C. Xiao, M. Azeem, and T. Mahmood, "Sentiment analysis on imdb using lexicon and neural networks," *SN Applied Sciences*, vol. 2, no. 2, pp. 1–10, 2020.

[129] M. Giatsoglou, M. G. Vozalis, K. Diamantaras, A. Vakali, G. Sarigiannidis, and K. C. Chatzisavvas, "Sentiment analysis leveraging emotions and word embeddings," *Expert Systems with Applications*, vol. 69, pp. 214–224, 2017.

[130] B. Everitt and A. Skrondal, *The Cambridge dictionary of statistics*, vol. 106. Cambridge University Press Cambridge, 2002.

[131] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, pp. 3111–3119, 2013.

[132] T. Mikolov, W.-t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pp. 746–751, 2013.

[133] D. Selivanov and Q. Wang, "text2vec: Modern text mining framework for r," *Computer software manual](R package version 0.4. 0). Retrieved from https://CRAN. R-project. org/package= text2vec*, 2016.

[134] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[135] K. Kim, J. Kim, J. S. Kim, F. Chazal, and L. Wasserman, "Efficient topological layer based on persistent landscapes," *arXiv preprint arXiv:2002.02778*, 2020.