Online at: https://researchtrendsjournal.com **ISSN No:** 2584-282X

INTERNATIONAL JOURNAL OF TRENDS IN EMERGING RESEARCH AND DEVELOPMENT

Volume 2; Issue 4; 2024; Page No. 76-82

Received: 14-05-2024
Accepted: 25-06-2024

The use of persistent homology in identifying and quantifying topological features of high-dimensional datasets

[1]Shivtirth Chaturvedi and [2]Dr. Uma Shanker

[1]Research Scholar, PK University Shivpuri, Madhya Pradesh, India
[2]Department of Mathematics, PK University Shivpuri, Madhya Pradesh, India

Corresponding Author: Shivtirth Chaturvedi

Abstract

In recent years, researchers in statistics and machine learning have begun to concentrate on topological data analysis, or TDA. Clustering and other methods that take use of data's geometry have been very useful in both theory and practice. The persistence homology approach is often used for TDA because it quantifies the importance of these invariants. Use of persistent homology is seen in several data visualization, statistical, and machine learning approaches. When combined with persistent homology and TDA, ML becomes more interpretable and generalizable than with only cutting-edge techniques. A variety of TDA applications and homology of persistence are investigated in my research. Since TDA has already been used to monitor intracellular particle motion, it makes sense that it might also be utilized to analyze the cellular cytoskeleton as a network seen in confocal microscopy pictures.

Keywords: Persistent, homology, topological features, machine learning, TDA

Introduction

Every cell in the eukaryotic cell lineage has protein filaments that make up the actin cytoskeleton. The actin cytoskeleton controls the form and mobility of cells, among other fundamental biological processes, in addition to acting as a scaffold for cells. Many complex physiological processes rely on these fundamental operations, including cell proliferation, migration, and motility.

Theoretically, myosin motor proteins and filament interactions jointly control actin cytoskeleton filament organization. The polar actin filaments are formed when globular actin proteins polymerize. A large number of proteins that bind to actin may potentially attach to actin filaments at different points all across the filament. Actin filaments may build and deconstruct in a spatiotemporal manner thanks to these binding proteins. Filaments grow into networks with several filaments and binding sites because of the intense cross-linking that the binding proteins cause. It is crucial to comprehend the mechanisms that control the arrangement of actin filament networks in order to comprehend certain cellular activities. The interplay between actin-binding proteins, certain filaments, and newly formed networks may provide an essential function in these interactions.

Creating a system for the categorization of actin networks is the objective of this effort. Confocal microscopy allows us to get the pictures of actin networks that pique our curiosity. The images still have a lot of noise, even though their high resolution (often around 2500x500 pixels with 0.04 µm2 pixels) allows them to distinguish features such as filaments that cross the focus plane, rounded edges of cells, contamination of the picture by surrounding cells, changes in microscopic settings, and conditions, and various other things. Thus, confocal microscopy offers a wealth of high-quality data, but it also includes a fair amount of noisy data. Therefore, a device that can withstand a wide range of noise levels is necessary for the automated analysis of these photos, free from researcher interventions (which would eliminate the possibility of introducing unanticipated bias).

Literature Review

Fujita, Takaaki. (2024) [1]. Examining the pathways, topologies, and features of networks made up of vertices (nodes) and edges is the main focus of graph theory, a foundational part of mathematics. The "graph width parameter," a crucial statistic in this area, measures the

greatest width across all cuts or layers in a hierarchical graph decomposition. Because of its versatility, tree-width in particular has been the subject of much study. We study these characteristics and their uses again in this paper, this time concentrating on tree-width and related graph width parameters as they apply to Bond Graphs, Factor Graphs and Graph Entropy.

Umeda, Y. & Kaneko, J. & Kikuchi, H. (2019) [2]. Interest in deep learning has recently skyrocketed and other machine-learning technologies, which has led to an acceleration in the commercialization of AI technology. But statistical data analysis is the foundation of machine learning, and it is well-known now that analytical procedures may lead to the loss of certain information contained in such data. Our new machine learning method, rooted to optimize the use of topological data analysis (TDA), utility of this kind of information by analyzing the "shapes of data." Learn about TDA, a novel approach to data analysis, in this study. Additionally, it details anomaly-detection technology and time-series deep learning, both of which are applicable examples of TDA; the latter is described in relation to an evaluation of bridge damage.

Huang, Dazhi & Xu, Pengcheng & Huang, Xiaocheng & Chen, Jiayi. (2024) [3]. Topological Data Analysis (TDA) has lately seen a surge in popularity for use in financial forecasting. But prediction outcomes are quite sensitive to the topological feature representations used, classification models used, and point cloud building methodologies used. The issue of how to categorize changes in stock indexes is tackled in this work. To start, we use three distinct approaches to build point clouds for stock indexes. Then, we extract topological properties from the resultant point using TDA clouds. To make sense of the data patterns, we calculate four topological characteristics, then we list all possible permutations of these features and feed them into six separate ML models. To get insights into the efficiency of different TDA configurations, we run execute index movement classification tasks on datasets such as CSI, DAX, HSI, and FTSE in order to evaluate the prediction accuracy of various TDA setups.

Leykam, Daniel & Angelakis, Dimitris. (2023) [4]. Methods for consistently and methodically "Shapes" of complex data sets may be constructed by topological data analysis. Physicists are showing an increasing amount of interest in topological data analysis due to its many potential uses in the data and biological sciences. Here we provide a brief overview on physics-related machine learning problems, including topological data processing, and problems with detecting phase transitions without supervision, and their potential applications. The paper concludes with a brief outline of potential future research paths.

Onyango, Allan & Okelo, Benard & Omollo, Richard. (2023) [5], Using AI and ML approaches, we conduct a comprehensive analysis of topological data (TDA) of the COVID-19 pandemic. Using huge data sets in Hausdorff spaces, we display the global distribution patterns of the epidemic at its height. Based on the data, it seems that the regions of the globe with the worst winters were the hardest hit.

Persistence homology and filament networks

A modification that exposes the hidden geometric aspects of our data is necessary to quantify the differences in filament networks. Constructing simplicial complexes using persistent homology as a standard procedure accomplishes this transformation. The starting points are the two-dimensional coordinates of the spots along the filaments that were sampled. Simplicial complexes, as mentioned in Chapter 1, allow connects the data space to a topological space, allowing for the computation of distances between collections of data points.

Data

Microscopy data: Each cell's microscopy data was shown as a single grayscale picture. The intensity of each pixel in a picture may be seen as a potential indicator of the existence of an actin filament in that specific area, as actin filaments glow. A topological transformation is necessary for investigating something which is homological to an action network.

Simulated data: Each cell's microscopy data was shown as a single grayscale picture. The intensity of each pixel in a picture may be seen as a potential indicator of the existence of an actin filament in that specific area, as actin filaments glow. A topological transformation is necessary for investigating something which is homological to an action network.

Persistence Homology

We use the method of building the Vietoris-Rips complexes on every dataset (actin network) to construct simplicial complexes. This involves centering a series of ϵ-balls with increasing radius ϵ at each data point, which may be a sampled pixel in image data or an actin bead in synthetic data. Each value of ϵ represents an unordered set of homological properties, known as a homology group, and the intersections of these ϵ-balls form simplicial complexes. By viewing the values of ϵ as a timeline, we may easily monitor the emergence and erasure of homological characteristics.

The finding and summary of the persistent homology of a simplified filament network. At $\epsilon = 0$, each of the filament network's sampled points is a linked component in and of itself. The merging of linked components starts as the ϵ-spheres expand. Several of the sampled points have already linked at $\Im = 0.3$ in the figure's first column. The persistence barcode below shows points that have died (merged into a greater connected component) by ending their bar. Their precise time of death is shown on the y-axis of the persistence diagram that follows, with a point at 0 on the x-axis. With $\epsilon \approx 1$, two holes will appear. In the first row, you can see the holes clearly section two of the diagrams, where the two holes-one larger and one smaller-are made apparent. Below each persistence barcode, we start to plot bars for the holes. To plot the holes in the two-dimensional diagram, we need to know when they died, which is why there are no records of them in the persistence diagram as yet.

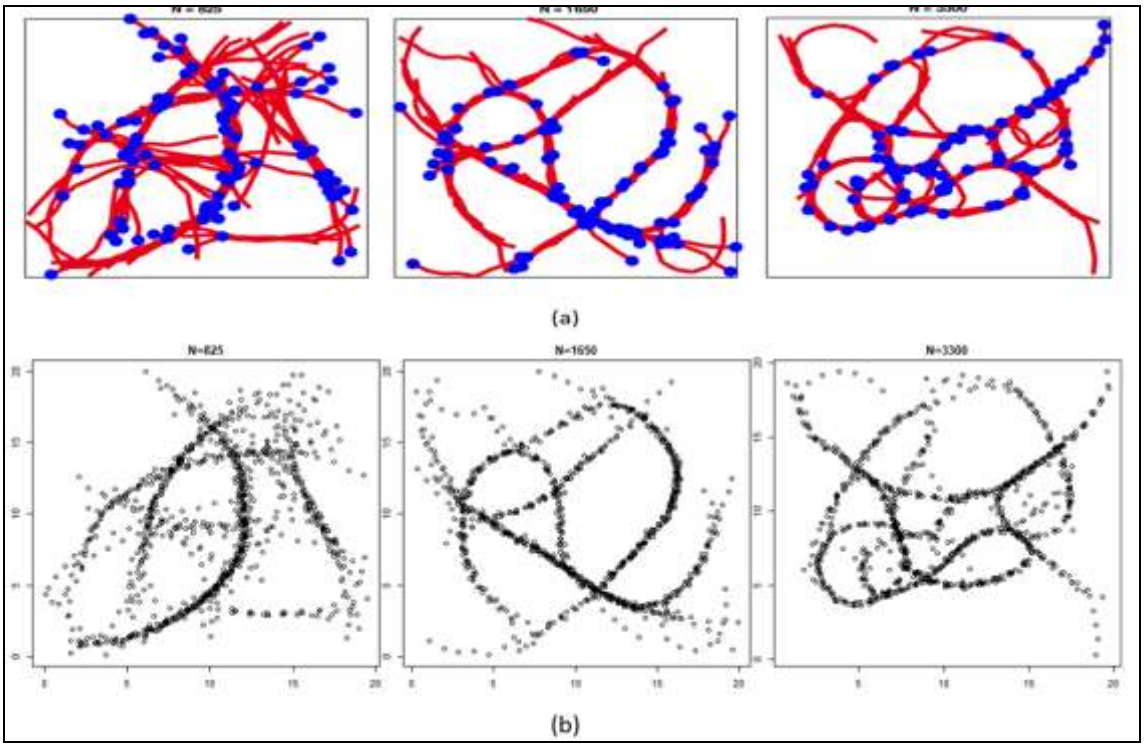

Fig 1: Networks of filaments. Within a 20 µm x 20 µm space, Panel (a) displays three filament networks that were produced by 825, 1650, and 3300 cross-linkers, respectively. The red lines depict the 100 filaments that make up each network. The filaments' barbed tips may be seen as blue spots. The actin beads that constitute the filaments seen in Panel (a) are shown in Panel (b).

Filament Network Classifier

The next step in classifying actin networks is to obtain persistence diagrams that match the interconnected nodes in these. In this chapter, we provide two approaches-one based on vectorization and the other on distance-that might be used to train a filament network classifier.

Distance-based network classifier

We want a measure of the dissimilarity in the space of persistence diagrams in order to compare any two of these diagrams. Both the Bottleneck and the Wasserstein distances may be used in TDA to determine the distance between persistence diagrams. Finding the best (cheapest) way to connect the dots in two persistence diagrams is what these distances are all about. A point in this made-up set corresponds to an off-carnality point, as they assume an endless number of points of multiplicity on the diagonal (where birth = death).

We include the dc distance into the existing Wasserstein and Bottleneck distances a novel metric that was suggested in and shown to be stable in to our set of chosen metrics. Particularly for homological properties that decay rapidly and could be deemed irrelevant in Wasserstain distance, the cardinality of a persistence diagram can potentially provide useful information in practical applications.

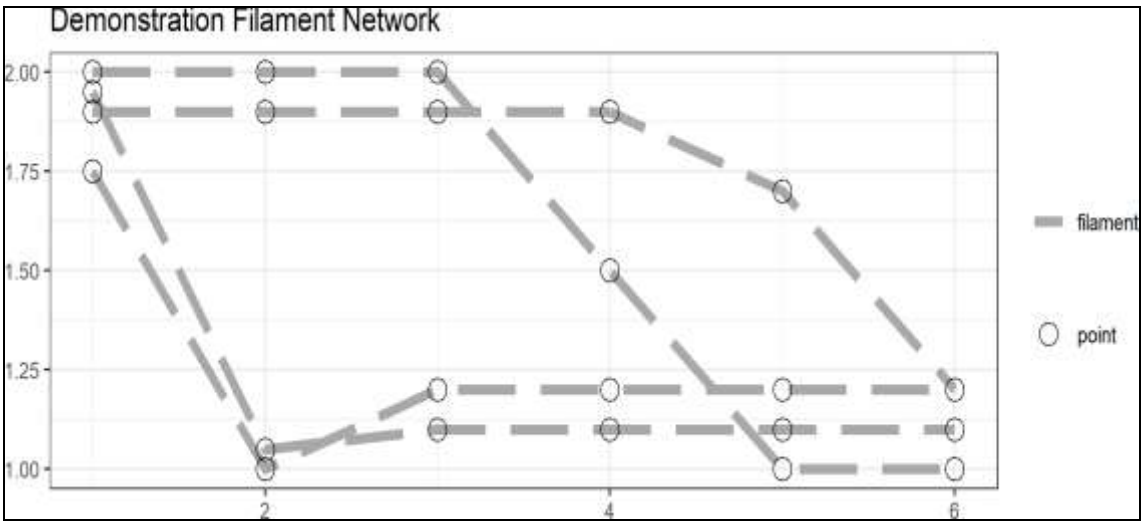

Fig 2: Network of demonstration filaments. There are three filaments in this network. A point cloud is generated by sampling points along the filaments; this cloud is then used to study persistent homology

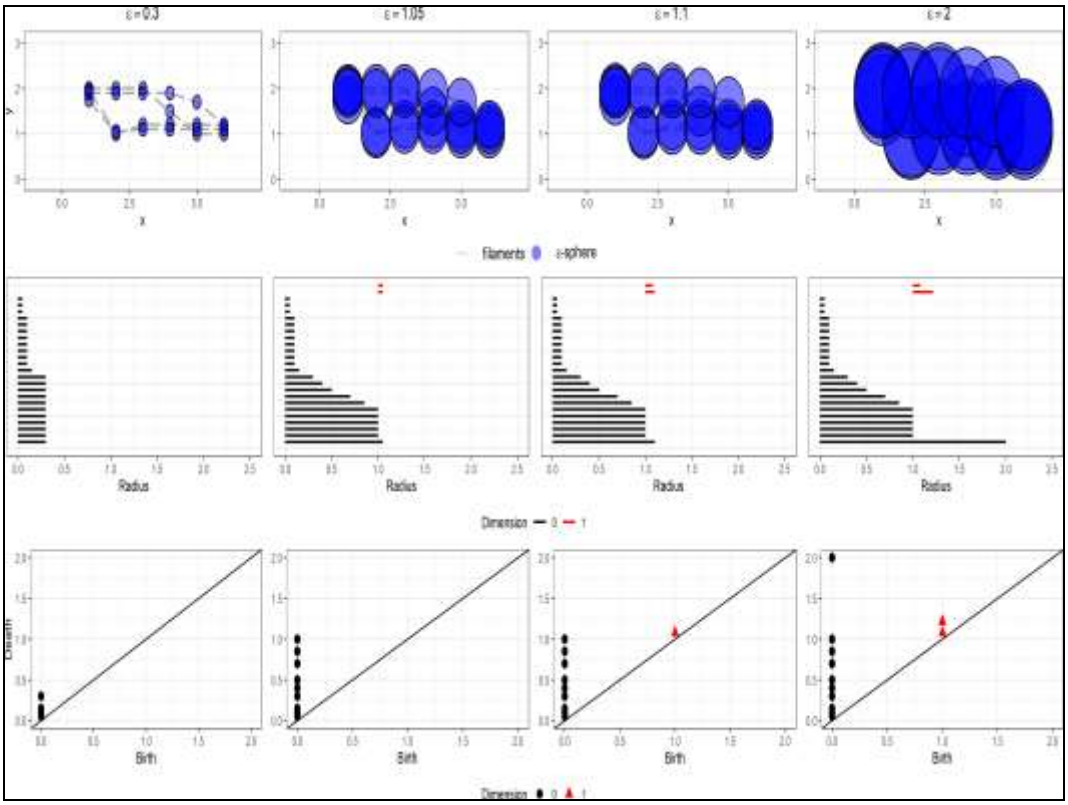

Fig 3: Analysis of the demonstration filament network's persistent homology (Figure 2). Around the places where the filaments were sampled, the increasing ϵ-spheres are shown in the first row of figures. The persistence barcode that corresponds to it is shown in the second row. The related persistence diagram is shown in the third row. In a right-to-left fashion, the columns follow the algorithm's development.

Definition 3.1. 1 Let D_X and D_Y be$_p$ two persistence diagrams with cardinalities n and m respectively such that $n \leq m$ and denote $D_x = \{x_1,..., x_n\}$, $D_y = \{y_1,..., y_m\}$. Let $c > 0$ and $1 \leq p < \infty$ be fixed parameters. The d^c distance between two persistence diagrams D_x and D_y is

$$d_p^c(D_x, D_y) = \left(\frac{1}{m}\left(\min_{\pi\in\Pi_m}\sum_{l=1}^{n} min(c, ||x_l - y_{\pi(l)}||_\infty)^p + c^p|m-n|\right)\right)^{\frac{1}{p}},$$

Where Π_m is the set of permutations of (1,..., m). If $m < n$, define $d^c(D_x, D_y) := d^c(D_y, D_x)$.

The difference in cardinalities between the two sets of points also determines a penalty term that is included into the dc distance when calculating the distance between points in two persistence diagrams that do not include the simulated points on the diagonal. In equation (2.1), the constant parameter c determines the penalization weight to be applied to the dc distance. A bigger penalization will be produced by higher values of c. The standard value for the parameter p is 2, which is the same as the distance between two points on Earth. For the simple reason that these numbers work well in practice, we usually assess c between 0 and 1.

Definition 3.2. 2 D stands for a set of related persistence diagrams. The direct correlation distance (dc) between a persistence diagram (Dx) and the set of persistence diagrams (D) for a certain β-dim homological feature (β = 0, 1, 2,...) is where the size of class D is |D|.

$$d_\beta(D_x, \mathcal{D}) = \frac{1}{|\mathcal{D}|}\sum_{D\in\mathcal{D}} d_p^c(D_x, D),$$

We may now construct the dc-based network classifier when the aforementioned preparations are finished. For every particular class of filament networks, is created using a different set of constraints for each of the K classes. Hence, there are K collections of persistence diagrams, whereby each collection stands for a distinct category of networks that were formed under different restrictions.

Algorithm 1: d_p^c-based network classifier

Let B is highest dimension of homological features under consideration.

1. Take the training set $T_1, T_2, ..., T_K$ from each class of diagrams $\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_K$.
2. For a new network with its corresponding persistence diagram D', compute

$$d(D', T_k) = \sum_{\beta=0}^{B} w_\beta d_\beta(D', T_k).$$

where $\sum_{\beta=0}^{B} w_\beta = 1$, and w_β determine how much β-dim homological feature is considered,

3. Assign D' a class label c' such that,

$$c' = \arg\min_{1\leq k\leq K} d(D', T_k).$$

Network classifier that does not rely on distance

Sub-sampling approach

To get more out of the given network data, our technique

used a number of pre-processing procedures instead of simply working on the actin beads' coordinates. In addition to geographical coordinates, the actin bead data also contained an index that indicated which filament was sampled. Thus, our sampled locations contain characteristics three variables: x, y, and f; x and y are plane coordinates, while f is the index in F's identity map. This is because we have one hundred filaments for every one of our networks {X1, X2...X150}.

The coordinates of each filament's (x, y) beads were connected using R's spatial package to build the filaments. Using the bead-coordinates to generate a corresponding set of lines, a single filament network was then constructed multiline object. Next, square grids with 200 × 200 cells were used to project these items with many lines. Then, we used the identity function to sample the multiline objects into grids for each network.

We randomly selected 1000 points from each network three times without replacement because to the exponential growth of the Vietoris-Rips complicated computation instead of calculating one complex per network. We get three sets of one thousand points each network from this.

Algorithm 2: Re-sampling Algorithm

Let $X_i \in \{X_1 ... X_{150}\}$ the set of simulated actin networks.

Let $a \in \{A_{i,(x,y,f)} ... A_{i,(x,y,f)}\}$ the set of points in X_i. a is a pair, $(i,(x,y,f))$, where i is an index to X_i, x and y are planar coordinates and f is an index to a filament of X_i, $F_{i,f} \in \{F_{i,1} ... F_{i,100}\}$, along which a lies. See Panel A of Fig. 2.4 for a graphical depiction of one X_i. For simplicity, we will describe how our algorithm is applied to X_1:

1. Reconstruct the paths of each $F_{1,f}$ by connecting $A_{1,(x,y,f)}$. See Panel B, Fig. 2.4.
2. We project F_1 onto a 200x200 square lattice with the same bounds as the given by the data. The cells, $c_{x,y}$ of the lattice take values $\{0,1\}$, where $c_{x,y} = 1$ if it is intersected by one of F_1 and 0 otherwise. See Panel C of Fig. 2.4.
3. Retain only $c_{x,y}$ where $c_{x,y} = 1$.
4. Draw 1000 random $c_{x,y}$ without replacement and call these the elements of $R_{1,r}$. Repeat this twice so that r indexes three new samples drawn from X_1. The three colors in Panel D of Fig. 2.4 denote the index r.
5. Generate three separate persistence diagrams, $D_{1,r}$, for the new samples $R_{1,r}$. Shown in Panel E of Fig. 2.4.

Thus, the number of networks included in the sample grew from 150 to 450. Following the identical steps as in Algorithm 1, we calculated persistence diagrams for these 450 networks and used them to build a classifier.

Classification Result

based classifier

There are three distinct types of filament networks included in our dataset. In order to create filament networks, various classes of proteins need differing amounts of cross-linking. There are fifty examples in each class. So, in all, 150 different filament networks are available.

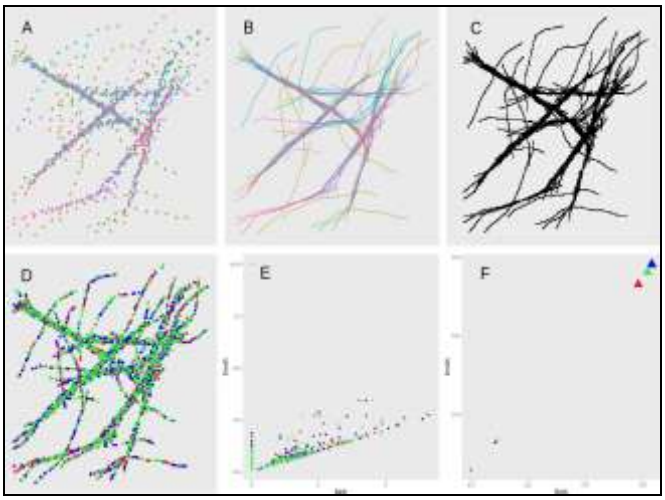

Fig 4: The following steps are illustrated: building a persistence diagram (E), rasterizing (C), resampling (D), and finally, vectorizing (F) the persistence diagram (B) using the provided point cloud (A). You can see the coloring in (B)-the filaments were built using the point cloud in (A) that was given with an index. A 200 × 200 raster (C) was used to project this. Using the x and y coordinates, a virtual reality complex was generated from a raster that had been randomly sampled into three sets of one thousand points (D) each. (E) and (F) use the same color scheme as (D) for their persistence diagrams and vectorizations to illustrate the variation in the characteristics of the vectorized persistence diagrams that results from this method.

Algorithm 3: Vectorization of persistence diagrams

We vectorize $D_{1,r}$ from Algorithm 2 by taking the mean *birth* and mean *death* of connected components and holes. These data are plotted in Panel F of Figure 3. For $D_{1,r}$ we have 3 row-vectors in the form:

$$v_{1,r} = \begin{bmatrix} \bar{x}(\text{Death of Connected components}) \\ s(\text{Death of Connected components}) \\ \bar{x}(\text{Birth of holes}) \\ s(\text{Birth of holes}) \\ \bar{x}(\text{Death of holes}) \\ s(\text{Death of holes}) \\ \bar{x}(\text{Birth of all features}) \\ s(\text{Birth of all features}) \\ \bar{x}(\text{Death of all features}) \\ s(\text{Death of all features}) \end{bmatrix}$$

Note: the mean $\bar{x}$ and standard error s of the birth of connected components would be 0 for all vectors, so these are not found in $v_{1,r}$. Repeat for all X_i. Our complete matrix of data has a final dimension 450×10.

The overall accuracy of the categorization was evaluated using 10-fold cross validation. R so that we could compare the classifiers. Ten sets of networks are randomly divided so that they cannot be joined. The training set consists of nine divisions, whereas the testing set uses the tenth. So that each division serves as a test set precisely once, we run the classification procedure ten times. When we average the accuracy across all partitions, we get the overall categorization accuracy rate.

non-distance based

A support vector machine (SVM) trained on the vectorized persistence diagrams and networked together to form the network classifier the 3-times re-sampled data. Once again, the SVM classifier's accuracy was evaluated using 10-fold cross-validation. For a fair comparison to the approaches without re-sampling, accuracy was tested on one of the three sub-sampled networks Ri,1 per filament network Xi. On average, the classifier had a 96.3% success rate. For these simulated networks, this level of precision was the maximum possible.

Comparison to other classifiers

Data transfer to a topological space and persistence diagram summarization still our results demonstrate that re-sampling may provide a more trustworthy filament network classifier. Compared to the Wasserstein-based classifier, the dc-based classifier had much superior results. No other popular classifier can compare to our dc-based and re-sampled classifiers, as well as our vectorized persistence diagram classifier. Based on these results, it seems that similar algorithms might be used to categorize microscopy data in the future.

Clustering of filament networks

Our research shows that using a small sample size for supervised classification without state-of-the-art accuracy could compare to may make diagnostics and inference troublesome in our classification findings. There is only one misclassification to look at in the classification result of each cycle when using cross-validation. Consequently, a trained eye would have a very hard time visually inspecting the misclassification and identifying a trend. Biologists who want to visually inspect information about persistence that is reflected back into the data space may find our alternative approach, which we can supply via unsupervised clustering, to be useful. Here, we investigate the result clusters using k-means clustering and attempt to learn from the incorrect cluster designations.

Instead of utilizing the subsampling method, we choose an R-value such that there is one vector vi,1 for every cell in our clustering application. We employ the first two main components after transforming our data into them, rather than clustering straight on the vis. A main component plot is shown in Figure 6. The cells labeled as mutant (M) and wild-type (WT) are colored differently. While the image does show class distinction, it also shows that the classes are not tightly packed around any particular points.

For each given number of partitions k, K-Means clustering seeks to divide our information in a manner that reduces the WGSS to a minimum. The Elbow Method is a heuristic that may be used to find the best value of k. With k ranging from 1 to K (where K << n), the Elbow Method calculates the WGSS conclusion from our data. Pick the least k so that returns on greater k are deemed too little. In order to minimize the WGSS and avoid overfitting, this heuristic was developed. For each value of k in the interval {1, 2...15}, we display the output WGSS in Figure 7. No significant drop in benefit as k increases is shown in this figure. The absence of obvious data division allows this to be explored.

Based on our experiment, k = 2 is a reasonable option for k, and we didn't uncover any compelling evidence to support any other value of k. So, we performed the clustering after dividing the data into two sets. The output groups are a mishmash of classes, meaning that our top accuracy result (80.9%), is not achieved with this strategy. In Figure 8, we display the data plotted with colored labels and shapes representing cluster assignments.

Table 1: Accuracy rate of classifiers

Classifier	Accuracy
SVM, re-sampled	96%
dc-based p	89%
Wasserstein-based, re-sampled	87%
Wasserstein-based	83%
Bayes Factor	83%
SVM PI	75%
Neural Net PI	71%
SVM Raster	65%
Random Forest Raster	55%

Fig 5: A picture of a WT cell in its raw form

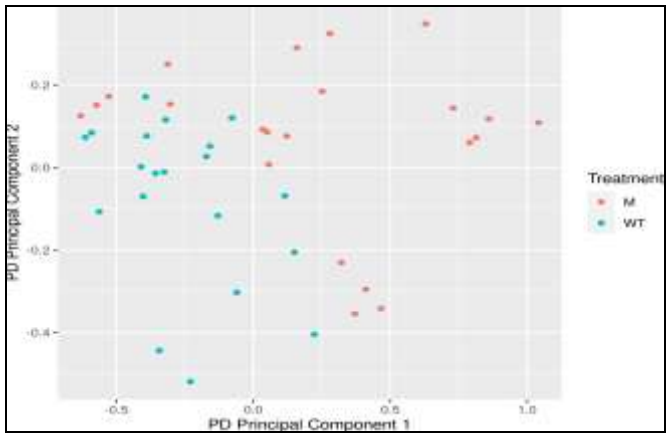

Fig 6: Main components of the cell's persistence data plot. Various treatments are used to colorize the points.

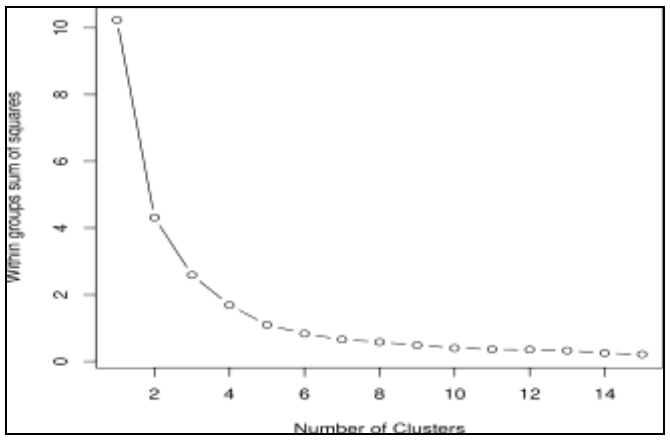

Fig 7: WGSS that uses a range of cluster sizes (k-values)

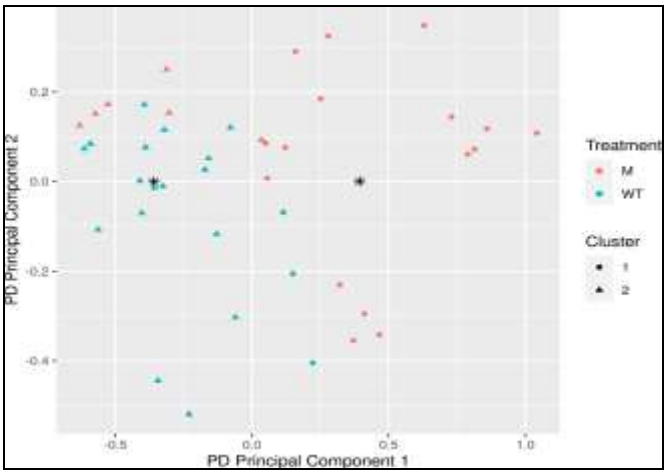

Fig 8: Main components of the cell's persistence data plot. The points are colored according to their treatment, and their shapes represent their k-means clustering class assignments. Asterisks denote the locations of the cluster nodes

Conclusion

We achieved extremely high accuracy in classifying simulated actin filament networks that were formed under three distinct beginning circumstances. To improve the accuracy of our categorization, we integrated TDA with a machine learning framework. Various distance-based classifiers were evaluated by us. Modern approaches to investigating actin network architecture are laborious, require extensive preprocessing and human intervention, and provide results that are very weakly connected to the dependent variables [10]. By separating a number of types of simulated cells, I proved that TDA is useful. In order to make these findings even better, I showed a new way to enhance TDA's data. I showed how to transfer approaches from synthetic data to microscope data analysis, and To prove my thesis, I examined cells that had and did not have a genetic knockout. This study utilizes persistent homology to examine actin networks for the first time. Not only that, but this method is the first of its kind to analyze filament networks entirely automatically. Our colleagues in the fields of biology and engineering will be submitting a more comprehensive proposal that will include our study.

References

1. Fujita T. A brief overview of applications of tree-width and other graph width parameters. 2024. https://doi.org/10.13140/RG.2.2.10489.28004.
2. Umeda Y, Kaneko J, Kikuchi H. Topological data analysis and its application to time-series data analysis. Fujitsu Sci Technol J. 2019;55:65–71.
3. Huang D, Xu P, Huang X, Chen J. Exploring applications of topological data analysis in stock index movement prediction. 2024. https://doi.org/10.48550/arXiv.2411.13881.
4. Leykam D, Angelakis D. Topological data analysis and machine learning. Adv Phys X. 2023;8:10.1080/23746149.2023.2202331.
5. Onyango A, Okelo B, Omollo R. Topological data analysis of COVID-19 using artificial intelligence and machine learning techniques in big datasets of Hausdorff spaces. J Data Sci Intell Syst. 2023;1. https://doi.org/10.47852/bonviewJDSIS3202701.
6. Carlsson G, Vejdemo-Johansson M. Topological data analysis with applications. 2021. https://doi.org/10.1017/9781108975704.
7. Ohanuba F, Ismail MT, Khan M. Topological data analysis via unsupervised machine learning for recognizing atmospheric river patterns on flood detection. Sci Afr. 2021;13:e00968. https://doi.org/10.1016/j.sciaf.2021.e00968.
8. Zia A, Khamis A, Nichols J, Tayab U, Hayder Z, Rolland V, Stone E, Petersson L. Topological deep learning: a review of an emerging paradigm. Artif Intell Rev. 2024;57. https://doi.org/10.1007/s10462-024-10710-9.
9. Patania A, Vaccarino F, Petri G. Topological analysis of data. EPJ Data Sci. 2017;6. https://doi.org/10.1140/epjds/s13688-017-0104-x.
10. Rieck B. Topology meets machine learning: An introduction using the Euler characteristic transform. 2024. https://doi.org/10.48550/arXiv.2410.17760.
11. Kvinge H, Wight C, Akers S, Howland S, Choi W, Ma X, *et al*. A topological framework to improve analysis of machine learning model performance. 2021. https://doi.org/10.48550/arXiv.2107.04714.
12. Montgomery RM. Cross-disciplinary applications of Riemannian and symplectic structures in machine learning and topological data analysis. 2024. https://doi.org/10.13140/RG.2.2.30084.95368.
13. Arfi B. The promises of persistent homology, machine learning, and deep neural networks in topological data analysis of democracy survival. Qual Quant. 2023;58:1–43. https://doi.org/10.1007/s11135-023-01708-6.
14. Gabella M. Topology of learning in feedforward neural networks. IEEE Trans Neural Netw Learn Syst. 2020;PP:1–5. https://doi.org/10.1109/TNNLS.2020.3015790.
15. Dey T, Mandal S, Mukherjee S. Gene expression data classification using topology and machine learning models. BMC Bioinformatics. 2022;22. https://doi.org/10.1186/s12859-022-04704-z.