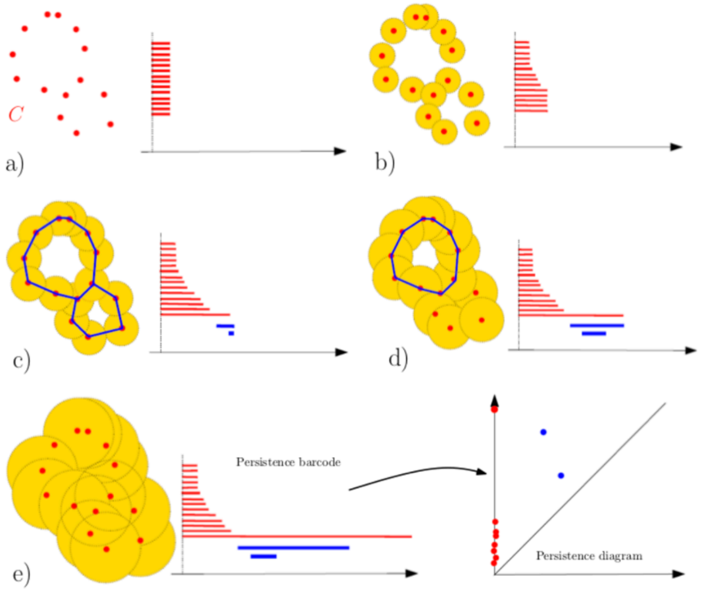
Topological data analysis (TDA) is a mathematical approach to understanding complex data sets that studies the topology and geometric structure of data to extract meaningful information. Specifically, TDA can provide extremely refined and often hidden structures in data, making it important in a variety of applications including bioinformatics, brain network analysis, and computer vision. There are many computational advantages of TDA, such as the prominent branch of persistent homology, a method which characterizes the shape of data using the appearance and disappearance of holes throughout a nested sequence of spaces. In a given nested sequence $X_1 \subseteq X_2 \subseteq . . . \subseteq X_n$, the inclusion $X_i \subseteq X_{i'}$ for $i \leq i'$ produces a linear map $H_k(X_i) \longrightarrow H_k(X_{i'})$ on the corresponding k-th homology. As the time parameter i increases, persistent homology tracks elements of $H_k(X_i)$, most often represented with a persistence diagram (PD) in the Cartesian plane R^2. Each point (x, y) in a PD corresponds to a feature appearing at scale x (birth) and disappearing at scale y (death), and has a persistence value of $(y - x)$. Points of high persistence in a PD are more important than those of low persistence, and points close to the diagonal are usually classified as noise while those farther away are more robust.

In order to use machine learning techniques such as SVM for classification of a given data set, one needs a vector representation for each PD. The method of converting a PD, say B, into a persistence image (PI) provides this bridge while still retaining most of the original information. This requires transforming B from birth-death coordinates to birth-persistence coordinates via the linear transformation $T\colon R^2 \longrightarrow R^2$ defined as $T(x, y) = (x, y - x)$, where $T(B)$ is the resulting transformed set. The points $(x, y - x)$ in $T(B)$ are then centered using a differentiable probability distribution $\varphi_u : R^2 \longrightarrow R$ with mean $u = (u_x, u_y)$ in R^2 and variance σ^2. A Gaussian distribution is usually used, defined by $g_u(x, y) = (1 / 2\pi\sigma^2)(e^{-[(x-ux)^{\wedge}2 + (y-uy)^{\wedge}2]/2\sigma^{\wedge}2})$. A nonnegative weighting function f: $R^2 \longrightarrow R$ is then fixed to produce the corresponding persistence surface $\rho_B(z) = \Sigma_{u \in T(B)} f(u)\varphi_u(z)$. The weighting function f is very important as it ensures a stable transformation. Common choices are a linear weight function or bump weight function which weighs points with high persistence very strongly. Lastly, the surface is reduced to a finite-dimension vector by overlaying it on a grid and integrating over each pixel to get the PI. The PIs serve as a useful method to combine PDs of different homological dimensions, say $H_0, H_1, ..., H_k$, into one concatenated vector that serves as the input for ML techniques.

In the example below we consider the filtration given by a union of growing balls centered on the finite set of points C.

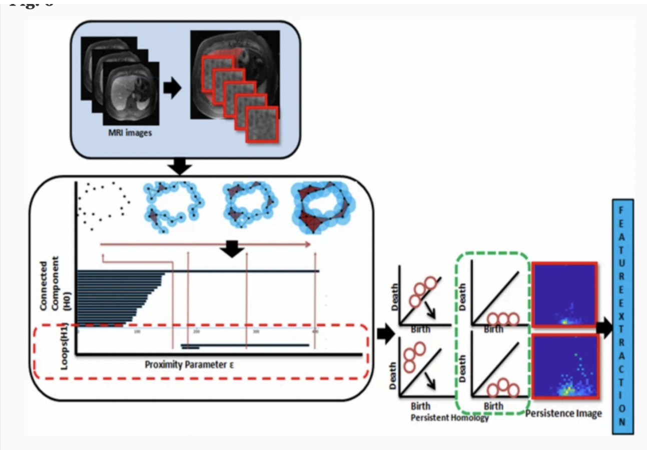


The above image shows an example of the persistence barcodes overtime and the resulting persistence diagram. As shown in figure a, the barcode starts off with n distinct components, where n is the number of initial points. As the size of the points increases, the number of components decreases and cycles begin to emerge until the point-cloud data becomes one component with no holes. After filtration is complete, the results are summarized in the persistence diagram, which compares the time of birth and death of each component.

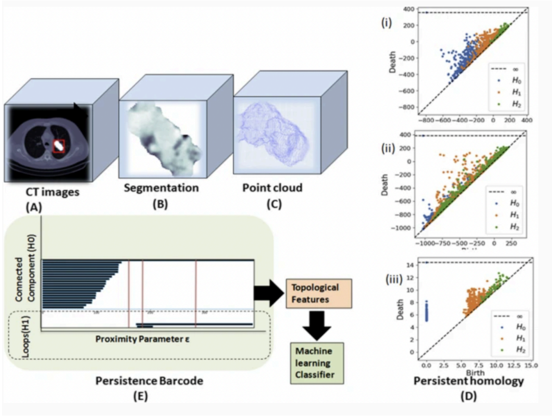
The current algorithms to calculate persistence diagrams and barcodes work by constructing filtration of a simplicial complex of the dataset. The function *f* runs in O(n^3) time, where n is the number of simplices, proving very inefficient. The vineyard algorithm takes a persistence diagram computed by function *f* and computes a perturbation *f'*, applying a sequence of updates to the filtration. Because the algorithm uses pre existing filtration, it is able to calculate the perturbations in O(n) runtime, significantly reducing computational cost and providing a more effective technique for filtration. Thus, the vineyard algorithm opens up opportunities for TDA in ML algorithms where it would otherwise be too computationally expensive.

In our project, we focused on TDA's applications in medical diagnoses. As an emerging state-of-the art technique, TDA has many applications in medical image processing and prognosis predictions. As an alternative to convolutional neural networks (CNNs), TDA in combination with ML techniques such as SVM is able to classify images and make predictions much more efficiently. Traditional CNNs use similar frameworks to TDA, extracting specific topological pixel features of an image. However, CNNs require higher volumes of training data compared to TDA. Additionally, TDA is able to work in higher-dimensional feature spaces — for example, in a spectral MRI or CT scan.

Below are examples of TDA's applications in high-dimensional medical imaging:



Workflow of algebraic topology-based machine learning with MRI imaging signal as input



TDA is also an extremely useful method for calculating time-series data consisting of a sequence of observations x_1, …, x_{t-1}, x_t, in which the order of observation matters. Real-world time series data often contains high levels of noise, rendering it difficult to detect similarities and homologies within it. With the application of arrhythmia detection, for instance, current machine learning techniques such as neural networks reduce the classification to a few specific types of arrhythmias which results in low model performance because of individual differences in real patient data. TDA, on the other hand, provides a powerful characterization of ECG signals and heartbeats for arrhythmia detection because it is extremely robust to individual differences in signal patterns. This characterization involves examining the persistent homology with a function *f* of the time-series encoded in a persistence barcode where the initial point of each heartbeat (represented in the barcode as with interval, *I*,) corresponds to a the creation of a new component valued α and the end point, valued α' corresponds to where this component merges with the subsequent one. To make these barcodes useful for machine learning techniques, they are converted in Betti curves, whose value for each α is determined by the number of intervals with that value α. These curves are time independent, meaning that they are stable and resistant to the re-parametrization of time and the rescaling of signal value, α. This is because the persistence intervals measure the height of signal peaks rather than their width. This solves the otherwise prominent issue of individual differences observed with other methods of handling time-series data. [1]

[1] https://arxiv.org/pdf/1906.05795

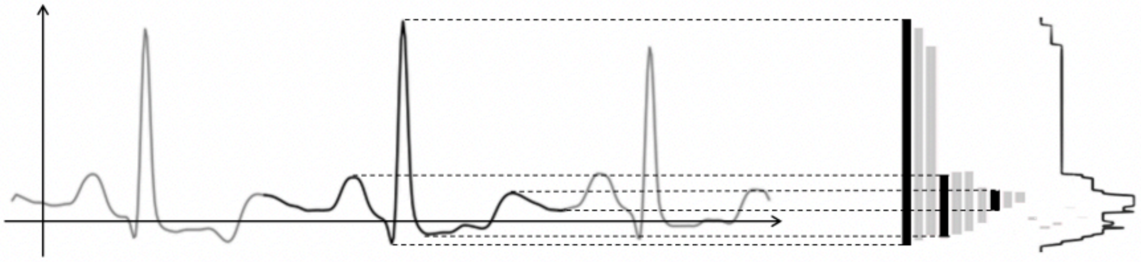
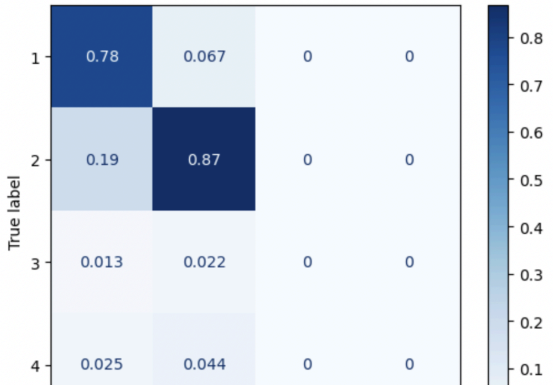
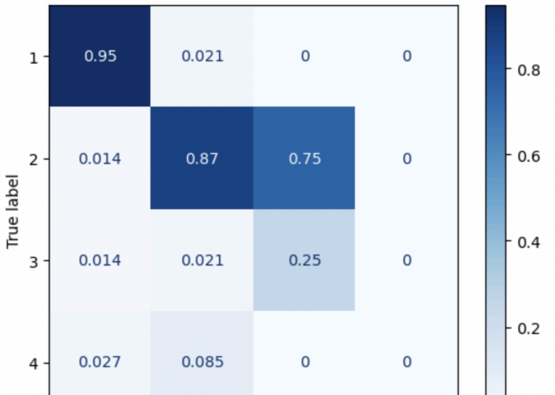


Figure 1: From three Consecutive Heartbeats to their corresponding persistence barcode and Betti curve

The above figure shows three heartbeats and the corresponding persistence barcode in the 0th dimension. Sharper and taller peaks correspond to longer-lasting connected components, while shorter or more spread out features do not persist for as long. As such, TDA is able to extract persistence features from the data that can be used to classify arrhythmias. Current literature focuses on the use of TDA in neural networks for classification, as well as multidimensional time scaling in R^3 to identify loops and higher-dimensional holes. For our project, we attempt to replicate this application on the ECG5000 dataset. We focus our analysis on persistent homology in the 0th dimension.

Our results on this arrhythmia dataset show that performing classification after having gone through a TDA filtration technique leads to improved accuracy scores as compared to performing no TDA on the dataset. For the initial binary classification using logistic regression between the A-fib group and atrial tachycardia group, the model performed with an 89.7% accuracy rate. However, we see that the performance improves for the modified set classification having an accuracy score of 97.4% while using logistic regression, 97.0% using a DTC with 12 leaves, and 100% using LDA. The modified dataset only focuses on the final 60 timesteps recordings, rather than the total 140 features. This modified set accounts for the fact that the most variance in heartbeat abnormalities comes from the final half of the data. Thus, we are able to focus on the more important features and achieve higher-accuracy scores. The models performed less well when comparing all four classes; the multiclass logistic regression accuracy score on the dataset with all 140 features was 80.0% and was 88.8% for the second half of the time series data. As shown in the confusion matrices, the model did best at correctly classifying the first and second labels, but poorly with the third and fourth labels. This is partially due to the fact that the third and fourth labels have significantly fewer occurrences — over 200 compared to less than



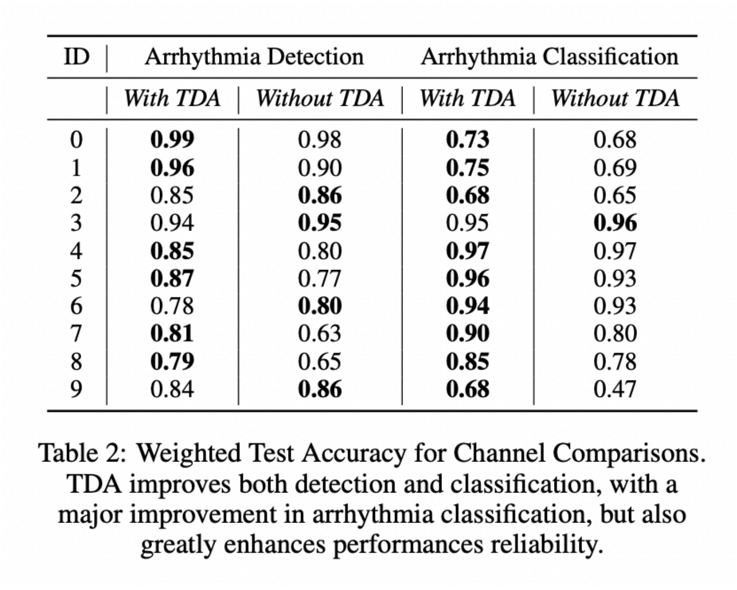


15. The DTC and LDA multiclass models performed similarly to the multiclass logistic regression model on the modified sets, having accuracy scores of 88.0% and 88.8%, respectively.

These discrepancies are particularly important to note and keep track of in the application of medicine or healthcare settings, as there often are two classifications occurring as we replicated in this project: the first one detects whether or not the heartbeat is normal, and the second actually classifies the type of abnormality on the arrhythmic heartbeats only. We would be interested to further explore how implementing the first type of classification affects the performance of models doing the second type, however this particular dataset included *only* arrhythmic heartbeats to begin with.

We also trained models on the raw training dataset without having priorly implemented TDA filtration, finding that the models perform better here than the one trained on only the first half of the TDA dataset but worse than those trained on the modified TDA dataset. The somewhat similar performance among models could suggest that this particular dataset was not significantly noisy or full of individual differences to begin with. Existing literature also suggests that TDA can increase the accuracy of detection and classification, remaining consistent with our findings.

	TDA: Binary (full set)	**TDA: Binary (modified set)**	**TDA: Multi Class (full set)**	**TDA: Multi Class (modified set)**	**Non TDA (full dataset only)**
Logistic regression	89.7%	97.4%	80.0%	88.0%	94.0%
DTC		97.0%		88.0%	94.0%
LDA		100%		88.8%	92.7%



ID	Arrhythmia Detection		Arrhythmia Classification	
	With TDA	*Without TDA*	*With TDA*	*Without TDA*
0	**0.99**	0.98	**0.73**	0.68
1	**0.96**	0.90	**0.75**	0.69
2	0.85	**0.86**	**0.68**	0.65
3	0.94	**0.95**	0.95	**0.96**
4	**0.85**	0.80	**0.97**	0.97
5	**0.87**	0.77	**0.96**	0.93
6	0.78	**0.80**	**0.94**	0.93
7	**0.81**	0.63	**0.90**	0.80
8	**0.79**	0.65	**0.85**	0.78
9	0.84	**0.86**	**0.68**	0.47

Table 2: Weighted Test Accuracy for Channel Comparisons. TDA improves both detection and classification, with a major improvement in arrhythmia classification, but also greatly enhances performances reliability.

Throughout this project, we gained experience in handling real-world data in the field of medicine as well as experience with a somewhat different type of dataset consisting of ECG signaling rather than images as we have been used to working with. We also learned a great deal of theoretical knowledge on algebraic topology, given that TDA and persistent homology has its background in this field, as well as time-series data. It was particularly interesting to build upon our class lecture on this topic, learning firsthand how the technique of TDA can be advantageous in working with this type of data. We also gained many additional skills in Python, learning how to use the gudhi and multiplier packages which are mainstream tools in the field of TDA for creating persistence diagrams and images.

References

"Topological Data Analysis for Arrhythmia Detection through Modular Neural Networks"
https://arxiv.org/pdf/1906.05795

"Persistence Images: A Stable Vector Representation of Persistent Homology"
https://jmlr.csail.mit.edu/papers/volume18/16-337/16-337.pdf

"Multiparameter Persistence Images for Topological Machine Learning"
https://web.ma.utexas.edu/users/blumberg/camera-ready.pdf