Topological Data Analysis for Time Series Classification

Abdrakhmanov Ildar [1] **Akhtyamov Timur** [1] **Mohamed Samir** [1] **Nikolaeva Anna** [1]

Abstract

Topological Data Analysis is an approach to analyze data using different techniques from topology. These techniques aim to extract fundamental qualitative properties, such as shape and connectivity in data. In this paper, we propose a universal approach for time series classification built on extracted qualitative features computed over persistence diagrams, Betti curves, and persistence landscapes obtained using techniques from the theory of persistent homology. Compared to standard classification approaches, the proposed methodology enables to classify time series, which have different recurrent behavior in the reconstructed phase space. Besides, it makes classification more robust to noisy and high-dimensional data. Thus, we assume it favors to classify data in which shape has meaning. Multiple experiments with time-series datasets confirm it.

1. Introduction

Topological Data Analysis(TDA) is becoming a popular branch in Data Mining. It appears in a lot of exciting applications in recent years. The crucial impact of TDA is an opportunity to extract important properties of qualitative nature from data. On the contrary, traditional data analysis provides tools to quantitatively learn hyperplanes from labeled examples in case of supervised classification learning. Still, it can't help to extract structural information, such as shape. In this work, we show that it facilitates the classification of time series. However, this is not the only case that benefits from a topological perspective. For instance, TDA is successfully used in detecting significant local structural sites in proteins (Sacan et al., 2007). Also, it is used in the finance field, helping in investment decision making (Goel et al., 2020). Any domain of applications of traditional data analysis could benefit from TDA also. TDA-based approach is more general and robust than constructing hyperplane in some metric space cause it makes it dependent on the metric chosen, which, for instance, can be corrupted by noise in the data (Tan et al., 2016). In our work, we concentrated on extracting different topological features from various topological summaries obtained with the help of persistent homology. The theoretical motivation for using persistent homology is provided in the preliminaries section. Some features extracted from persistent diagrams that we used are described in (Pereira & Mello, 2015). The main contributions of this report are as follows:

- Implementation of a general pipeline for extracting various topological features for time series classification datasets from UCR Time Series Archive (Dau et al., 2018).
- Training different classification algorithms on obtained topological features.
- Tuning hyperparameters of classifiers and Taken's embedding parameters to achieve the best quality possible on extracted features.
- Imputation study of which type of features are better for which particular dataset from UCR Time Series Archive.
- Performance ranking of 6 supervised classification algorithms with and without the deep generative model of Variational AutoEncoder (Kingma & Welling, 2013) as well as shallow artificial neural network.
- Comparing the performance of the classifiers with the benchmark algorithm(1-NN Euclidian distance).

2. Preliminaries

We provide an introduction to the notation and key concepts of topology, persistent homology, and time series analysis (Pereira & Mello, 2015). Topology studies only properties of geometric objects which do not depend on the chosen coordinates, but rather on intrinsic geometric properties of the objects. Consider a data set X, topology on this set is a subset $T \subseteq 2^X$ such that following properties are satisfied:

[1]Skolkovo Institute of Science and Technology, Moscow, Russia. Correspondence to: Abdrakhmanov Ildar <Ildar.Abdrakhmanov@skoltech.ru>, Akhtyamov Timur <timur.akhtyamov@skoltech.ru>, Mohamed Samir <samir.mohamed@skoltech.ru>, Nikolaeva Anna <anna.nikolaeva@skoltech.ru>.

Final Projects of the Machine Learning 2020 Course, Skoltech, Moscow, Russian Federation, 2020.

1) if $S_1, S_2 \in T$, then $S_1 \cap S_2 \in T$; 2) $\emptyset, X \in T$; 3) if $\{S_i | i \in I\} \subseteq T$, then $\cup_{i \in I} S_i \in T$; This means that topology is nothing but a system of subsets that determines the connectivity of a data set. The topological space $\mathbf{X}$ is the pair of a set X and topology T. In other words, the qualitative properties of topological space $\mathbf{X}$ is described by its decomposition into path-connected components. Any metric space is also a topological space, which is given by a set X and some metric function d.

To introduce persistent homology, we need to define a simplicial complex first. Simplicial complex $\mathbf{K}$ is a finite union of simplexes in R^N satisfying that every face of a simplex in $\mathbf{K}$ is in $\mathbf{K}$ and that the non-empty intersection of two simplexes in $\mathbf{K}$ is a face of each. In simple words, simplicial complex is a structure composed of points, line segments, triangles, and their n-dimensional counterparts. More intuitively, a 0-simplex is a vertex, a 1-simplex is an edge, a 2-simplex is a triangle, a 3-simplex is a tetrahedron, and so forth. An example of simplicial complex is illustrated in Figure 1. Persistent homology is a technique used to compute

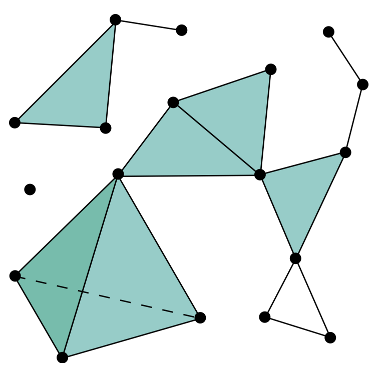

Figure 1. Simplicial 3-complex

topological features of a space represented as a simplicial complex at different spatial resolutions. The wider the range of spatial resolutions, the more persistent features can be extracted. These features are holes that appear and disappear, as simplicial complexes are created. The sequence of birth and death of n-dimensional holes is called a filtration. In our work, we detected only 0- and 1-dimensional topological features(holes). There are multiple ways to implement the filtration. In our project, we used Vietoris-Rips complexes that can be added to the filtration at some fixed value of filtration parameter if the distance between points in a simplex is less than this value. This variant of filtration is computationally intensive because the number of complexes is exponential in the number of points as it uses all the points. However, it's the most straightforward and accurate way. A common way to visualize topological holes is persistence diagrams(PD). The example of the persistence diagram is shown in Figure 2. Points on the persistence diagram corresponds to the n-dimensional holes. The abscissa indicates

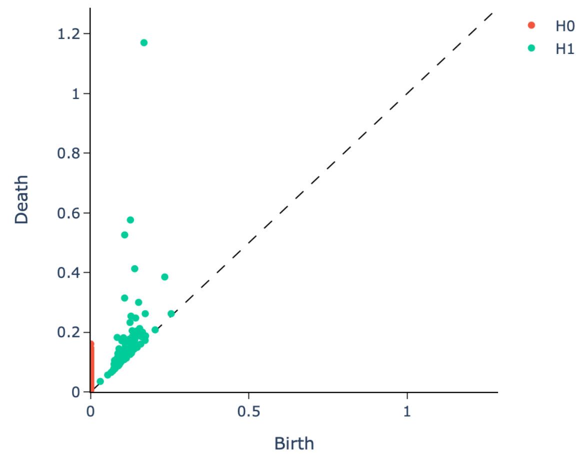

Figure 2. Persistence diagram

the birth and ordinate indicates the death of a particular hole. All points on the diagram are above the diagonal, and the most persistent holes are the farthest from the diagonal. Such holes are the most important ones. Holes on the diagonal have zero lifetime and do not contain any vital structure in data. Some points that are close to the diagonal may appear due to noise and some inconsiderable relationships in the data set. We used persistence diagrams as one of the sources of features used as input data to machine learning algorithms. Other sources of features and the approach to extract these features is described in section 4.

In this project, we deal with time-series data. We can't construct the Vietoris-Rips complexes using time series as input data. But we can represent them as a time series of the high-dimensional point clouds using Takens's embedding theorem (Takens, 1981). It's based on a time-delay embedding technique named after F. Takens. Given a discrete time series $(X_0, X_1, ...)$ and a sequence of evenly sampled times $t_0, t_1, ...$ we can extract a set of d-dimensional vectors of the form $(X_{t_i}, X_{t_i+\tau}, ..., X_{t_i+(d-1)\tau})$ for $i = 0, 1, ...$ This set of vectors is called a point cloud. Here, τ is a time delay, d is a dimension of the embedding. The upper bounds of this parameters we tuned to obtain the most relevant features from the time-series data sets. To calculate Takens's embeddings, we used a high-performance topological machine learning toolbox in Python called Giotto-tda[1]. The parameters of the embeddings are optimized as follows: optimal time delay is found by minimizing the time-delayed mutual information, embedding dimension is selected according to heuristics from (Kennel et al., 1992). In the next section, we review the recent and state-of-the-art approaches in TDA used for time series classification and beyond.

[1] `https://github.com/giotto-ai/giotto-tda`

3. Related work

Due to the active development of TDA (Frédéric Chazal, 2017; Chengyuan Wu, 2019), we have recently gotten several powerful tools to improve the quality of machine learning models by applying linear-size approximations to the Vietoris–Rips filtration, computing topological persistence for simplicial maps, etc. Using the topological analysis, sever studies have been provided, demonstrating its efficiency in tasks of time series prediction. In the article (Pereira & Mello, 2015), researchers particularly propose the generation of new features by taking the persistent homology described above. The presented pipeline demonstrated high accuracy on various UCR datasets. However, there were datasets on which pipeline performed not effective enough. Therefore, there is a space for improvement by, for instance, adding new features or incorporate sliding windows in the approach. In other works (Lee M. Seversky, 2016; Yuhei Umeda, 2019), similar ideas were presented. The significance of the use of sliding window transformations and the dependence of accuracy on its size were also shown. The approach targeted at constricting kernels directly from the persistent diagram for kernel-based models. Moreover, the ability to detect outliers and anomalies was confirmed as well. The paper (Umeda, 2017) reveals the potential of the Betti sequence generated from a quasi-attractor and its suitable usage with the 1-CNN model. However, the main drawback of the technique is a long computational time. Summarizing all the proposals, we have combined several ideas and made some modifications. The detailed description is presented in the next sections.

4. Algorithms and Models

4.1. Topological Features

In our project, we implemented the universal pipeline for extracting topological features and evaluating classification algorithms on these features. It is summarized in Figure 3. All source code is available on GitHub [2]. The first step of the pipeline is to construct the Takens's embeddings from univariate time-series classification datasets from UCR repository[3]. We tuned the upper bound of parameters of embeddings on the stage of classification algorithms evaluation. After this preprocessing step, we computed the Vietoris Rips complexes from the constructed embeddings to discretize the space and obtain the connectivity information. The maximum homology dimension of holes we computed is 1. In the third step of our approach, we obtained various topological summaries, such as persistence diagrams, Betti curves, and persistence landscapes.

[2]https://github.com/SamirMoustafa/Time-Series-Classification

[3]http://www.timeseriesclassification.com

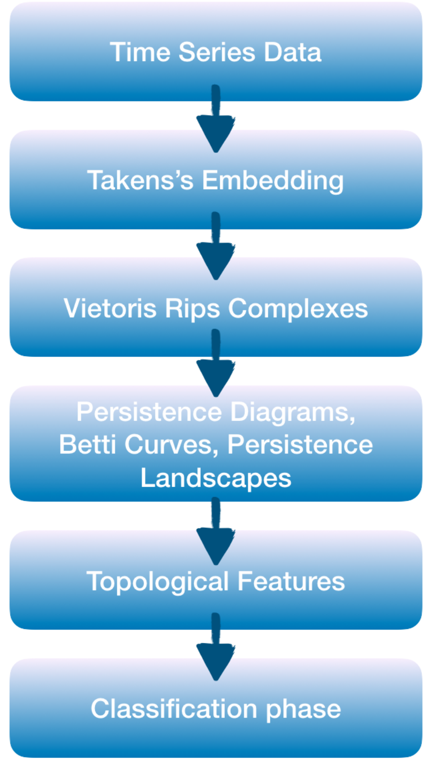

Figure 3. TDA pipeline for time-series classification

Persistence Diagrams

After computing the Rips Filtrations, we extracted six topological features from persistent diagrams. Five of them were proposed in this article(Pereira & Mello, 2015). The last feature was engineered to facilitate the extracting of important topological information. They are as follows:

- Number of holes for each dimension. Note that 0-dimensional holes indicate connected components along the filtration.
- Maximum hole lifetime in each dimension. It helps to find the hole which is the farthest from the diagonal and thus, the most persistent one. This feature can be described by the following equation: $\max_d = \max_{\pi_i \in}(d_i - b_i)$, where d_i and b_i are death and birth of a particular n-dimensional hole.
- Number of relevant holes. It is a fraction of points that which have a lifetime greater than lifetime of the most prominent hole scaled by the ratio. In our case we chose ratio $= 0.7$. This feature is important because some points from that are far from the diagonal could bear some important shape-based qulitative information. The feature is summarized in the equations below: $f(\text{life}, \max_d, \text{ratio}) = 1$ if life $\geq$ ratio$\cdot \max_d$ $f(\text{life}, \max_d, \text{ratio}) = 1$ if life $<$ ratio$\cdot \max_d$

- Average lifetime of all holes in dimension d. The equation which is used to compute this feature: $avg_d = \frac{\sum_{i=1}^{n}(d_i - b_i)}{n}$. It describes all the holes in the dimension d. Small number indicates that there are no prominent holes in that dimension for a particular data set.

- Sum of all lifetimes(integral of the Persistence diagram graph). It can be computed using equation below: $\sum_d = \sum_{i=1}^{n}(d_i - b_i)$

- Average number of simultaneously existing holes. This is the average number of holes that exist simultaneously for each dimension. It can help to describe how sparse the holes are distributed for each dimension. If we represent the persistence diagram as a set of sorted segments, which is also known as persistence barcode, this feature can be calculated as an average number of intersections of those segments.

Persistence Entropy

Persistence entropy is a measure of the entropy of the points in a persistence diagram. To define it formally consider the persistence diagram $D = \{b_i, d_i\}_{i \in I}$. The persistence entropy of D is defined as $E(D) = -\sum_{i \in I} p_i \log p_i$, where $p_i = \frac{d_i - b_i}{\sum_{i \in I}(d_i - b_i)}$. In the results section it was shown that this feature plays an important role when used as an input data for classification algorithms.

Betti Curves

Two topological features were extracted from this topological structure. The example of betti curves for one of the time series from a particular dataset is shown in Figure 4. The ordinate of this graph is the betti number, and the abscissa is a filtration parameter. Betti numbers are used to distinguish topological spaces based on the connectivity of n-dimensional simplicial complexes. The topology of an object is described by a set of Betti numbers, each number expressing the number of holes an object contains in n-dimensions.

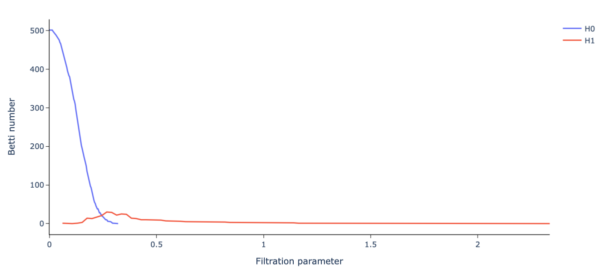

Figure 4. Betti curves

- Sum of Betti numbers. This feature can be considered as an integral of Betti curves for each dimension. It is introduced to describe the structure of Betti curves.

- Filtration parameter at maximum Betti number. This feature was engineered under the assumption that the configuration of the current filtration state when the Betti number is maximal may carry valuable topological insights.

Persistent Landscapes

Persistent landscape is a continuous function: $\lambda : N \times R \to R \cup \{+\infty\}$. Let $D = \{b_i, d_i\}_{i \in I}$ be a persistence diagram. Its associated persistent landscape is defined by letting $lambda_k(t)$ be the k-th largest value of $\min_{i \in I}\{t - b_i, d_i - t\}_+$. Intuitively, we can describe the graph of the persistence landscape by first joining each of the points in the multiset to the diagonal via a horizontal as well as a vertical line, then rotating the figure 45 degrees clockwise, and rescaling by $\frac{1}{\sqrt{2}}$. The example of calculated persistent landscape is illustrated in the Figure 5.

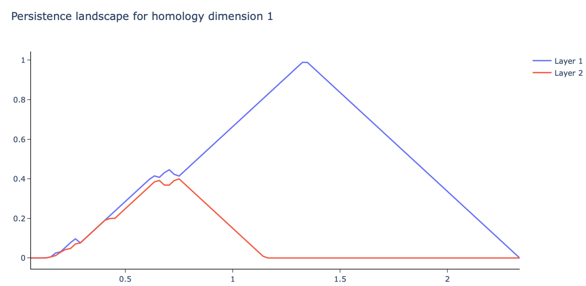

Figure 5. Persistent Landscape

The only feature we extracted from the landscapes is the average value of the persistence landscape. It can help to describe the structure of the persistence landscape diagram, which can be informative. It's essential to notice that in almost all of the studied cases, only the persistence landscape for the homology dimension zero is informative, so we consider this feature only for the zero homology dimension. When all topological features are extracted we build a new dataset and use it as an input data to multiple classification algorithms.

4.2. Dimensionality Reduction

4.2.1. Theoretical Motivation

Dimension reduction is a key concept where we map the data from a space to another space since the second space has dimension less than the original space to be more formally we can write to be like that.

$$X_m = \{x_1, x_2, .., x_m\} \in R^d$$
$$h : X_m \to Z_m; Z_m \in R^p, d > p$$

So the h maps from the R^d to the R^p space, previously dimension reduction techies used in visualization, compression, and estimation of the distribution, and recently with neural networks, we can generate data using generative adversarial networks (GAN) (Goodfellow, 2014).

For our case, we use it with neural networks to reduce our problem dimension to be in lower separable space using Variational Auto-Encoder (VAE).

4.2.2. Variational Auto-Encoder

One of the most key challenging in dealing with time series is extract is deterministic behavior in lower dimension space, So we define a general dynamic model for all the datasets that we have using convolution and de-convolution and fully-connected layers which can adapt itself to take the input and reconstruct it again using the latent space (Z), batch size (C) and the dimension of the data, describing it formally we can define our model like below.

Layer	input	output	kernel	stride
conv 1	1	C	4	2
conv 2	C	$2C$	4	2
mu	$d1.C$	Z	-	-
sigma	$d1.C$	Z	-	-
sampler	Z	$3C$	-	-
deconv	C	1	$d2$	3

Table 1: The dynamic architecture parameters depending on the input dataset, where C is batch size, Z is the latent or embedding space, $d1$ and $d2$ is a parameters extracted from the dataset dimensions

As shown in table architecture of the model have three fully connected layers at the bottleneck which can be visualize like the Figure 6.

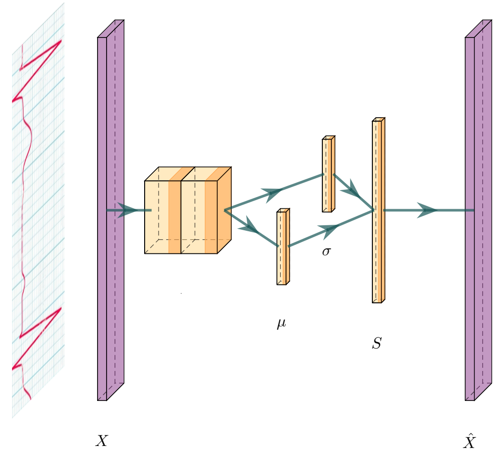

Figure 6. Variational Auto-Encoder Architecture: with input (X) and output ($\hat{X}$), and three fully-connected layers works as mean, variance, and sampler estimators (μ, σ, and S), and two convolution layers in the Encoder and one deconvolution layer as Decoder where the number of channels is detriment by the input dimension.

5. Experiments and Results

5.1. Time Series Data

To test the proposed methodology we run all of 128 univariate time-series classification datasets from the UCR repository through all stages of our pipeline. The UCR collection contains a variety of both multiclass and binary time-series classification problems. The smallest dataset contains forty time-series while the largest one contains 50000 examples. The most computationally-intensive part of evaluation is computing simplicial complexes using Vietoris-Rips filtrations this work is done in a parallel way where we do it for persistence diagram in extracting embeddings and also runnig a list of models done in a parallel way such that every model train and predict on parallel thread, where it's run on server with 80 cores and Nvidia Titan RTX, the models trained and evaluated on the cores and the VAE and neural network trained and evaluated on the GPU.

5.2. Classification algorithms and metrics

To evaluate the impact of our approach we considered four different classification algorithms with and without the variational autoencoder and one shallow neural network with variational auto-encoder as an additional component after the encoding part. Among those four are CatBoost, XGBoost, Support Vector Machine (SVM) and K-Nearest Neighbours (KNN) classifier. We used accuracy metric over which we optimized the hyperparamets of the classifiers and recorded accuracy for each possible pair: classifier - dataset.

Since we were analysing different models, we used a 2-folds cross-validation technique with stratification and shuffling to find best parameters for each of the classifiers. Those parameters are:

- SVM: regularization parameter ($-2E10$ to $-2E5$), kernel type (linear, RBF, polinomial, sigmoidal)
- XGBoost: maximal depth (2, 10, 15, 20, 25, 30, 35, 40, 45, 50, 70, 100, 120, 150), number of estimators (20, 50, 100, 150, 200, 25)
- KNN: number of nearest neighbours (3, 5, 7, 11)

Table 1. Classification metrics CatBoost algorithm on *TwoLeadECG* dataset

METRICS	RESULTS
ACCURACY	92.4
RECALL	92.4
PRECISION	92.5
F1 SCORE	92.3
MCC	0.85

- CatBoost: maximal depth (2, 10, 15, 20, 25, 30, 35, 40, 45, 50, 70, 100, 120, 15), number of estimators (20, 50, 100, 150, 200, 250), number of early stopping rounds (2, 5, 8, 10, 50, 200)

Also, for some of the datasets we also tried to tune parameters of the Takens's embeddings: embedding dimension and time delay. The reason for that will be described in Section 5.3. By default, we used embedding dimension equal to 2 and time delay equal to 3.

For the neural network we use fully-connected 4 layers, drop-out and batch normlization to takes the data from it's latent space (Z) and tried to fit it to the target values.

Furthermore, we calculated other metrics which are more informative for measuring the perfomance of the classifier. The following metrics were calculated:

- Accuracy: $Acc = (TP + TN)/(TP + TN + FP + FN)$
- Recall: $R = TP/(TP + FN)$
- Precision: $TP/(TP + FP)$
- F1-Score: $F1 = 2TP/(2TP + FP + FN)$
- MCC: $\frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$

,where MCC is a Matthews correlation coefficient, which measures the overall amount of agreement between the predictions and the ground truth.

For instance, consider *TwoLeadECG* dataset which is an ECG dataset taken from physionet by Eamonn Keogh. 1-NN Euclidian distance algorithm scored 74.7% on this dataset. CatBoost algorithm trained on topological features achieved accuracy of 92% on this dataset. The calculated metrics for this case are summarized in the table 1.

5.3. Imputation study

In this section we are going to figure out the impact of proposed features on the performance, along with understanding, for which type of datasets the proposed approach can be successful or not.

Of course, the influence on different features on the final performance is highly depend on dataset. Here we consider to look at the example with some datasets. Let's look at *GunPoints* dataset from UCR collection that is a binary classification problem. We achieved 100% on train and 90% accuracy on test with XGBoost classifier. Removing all of our own proposed features and leaving only Pereira's features will decrease the best accuracy on test set to 86%. In order to figure out more representative influence of features, we obtained feature importance diagram for XGBoost classifier, shown at Fig. 7, and for Logistic Regression that produced 98% train and 85% test accuracy, shown at Fig. 8.

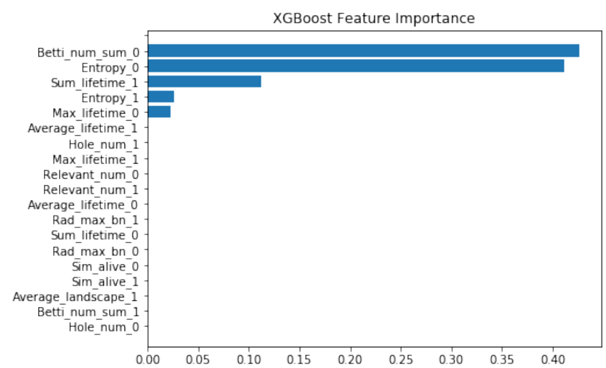

Figure 7. Feature importance with XGBoost for *GunPoint* dataset

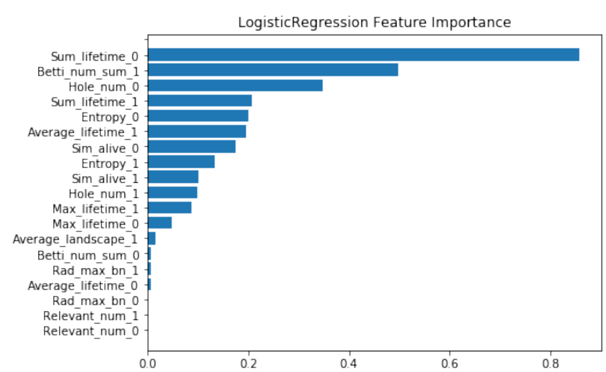

Figure 8. Feature importance with Logistic Regression for *GunPoint* dataset

Now let's take a look at multiclass dataset, for example, *Trace*. We achieved 100% train and 99% test accuracy with XGBoost and 99% train and 99% test accuracy with Logistic Regression. The corresponding feature importance diagrams are shown at Fig. 9 and Fig. 10. These obtained results shows that for that two datasets Persistence Entropy, sum of holes lifetime and sum of Betti numbers seem to be a strong and informative feature, as we expected while feature engineering.

Still, the unclear point is how to estimate for given dataset will topological features show good performance on it? Or,

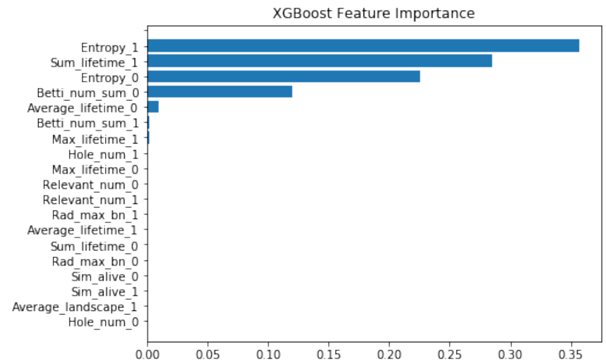

Figure 9. Feature importance with XGBoost for *Trace* dataset

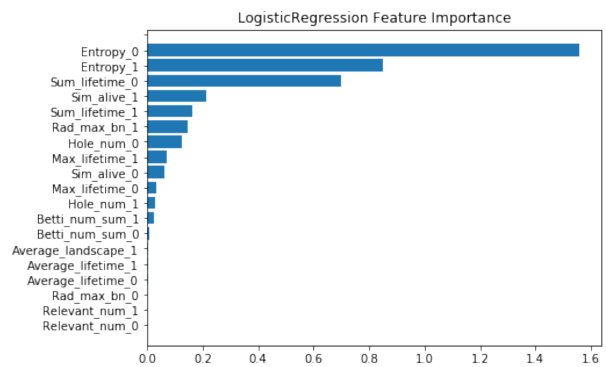

Figure 10. Feature importance with Logistic Regression for *Trace* dataset

in other words, for which type of datasets topological features will potentially work well and why? In order to answer this question we should remember that topological features are correspond to topological properties of given data like shape, connectivity and etc. Since we obtain Persistence Diagram from point clouds that produced by Takens's embeddings, let's look at some of that produced point clouds. The sample point clouds for two classes of *Trace* dataset are shown at Fig 11. and Fig 12. The difference for point clouds in this dataset is significant, thus, from topological point of view, those classes of time series are very "far" from each other and thus they are easily separable. Obtained results for this dataset proves that.

Now let's look at sample point clouds for *ECGFiveDays* dataset, shown at Fig. 13 and Fig. 14. We can see that in this case the shape look quite similar, the main difference here is "scale" of the loop. In other words, those two classes are quite similar from topological point of view, and thus we expect that topoligcal features won't be able to produce good classification quality. Actually, the best achieved quality for us for this dataset is 100% train and 68% test accuracy for this dataset.

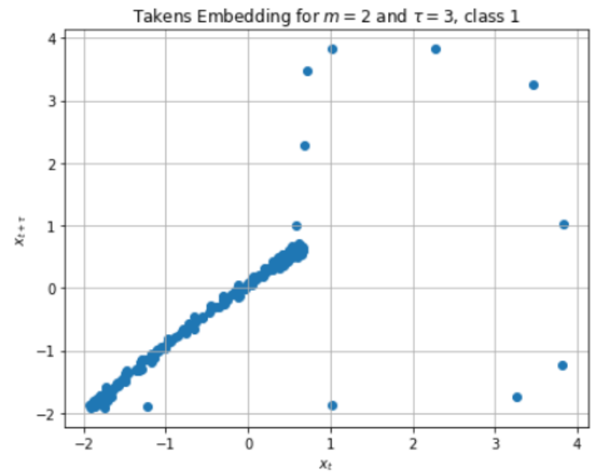

Figure 11. Sample point cloud from Takens's embedding of class 1 for *Trace* dataset

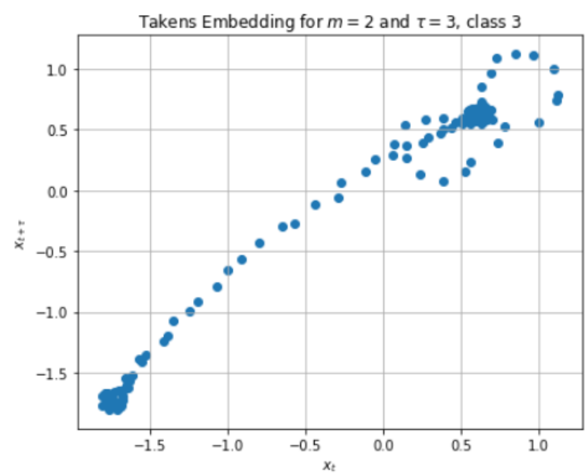

Figure 12. Sample point cloud from Takens's embedding of class 3 for *Trace* dataset

From this we can propose following conclusion. The potential performance of topological features highly depend on initial data structure and point clouds that it can produce. If point clouds have distinguishable shapes and different combinations of holes, loops, etc. for different classes, then we can expect that topological features produce good quality as they will capture those differences. Otherwise, if those forms are not so distinguishable, and the most of the difference in scales and distances, metric-based method will probably outperform topological features based approach. Still produced point clouds can be tuned with parameters of Takens's embeddings, but for the evaluation the Persistence Diagram is also needed, and computing it can take a lot of time for the dataset, and it makes such parameter tuning quite complicated and computational-intensive task.

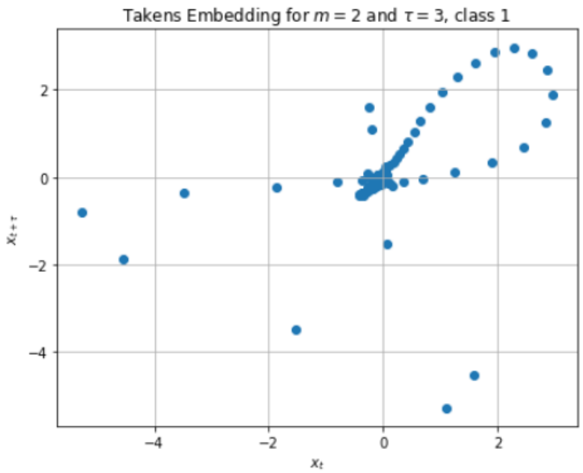

Figure 13. Sample point cloud from Takens's embedding of class 1 for *ECGFiveDays* dataset

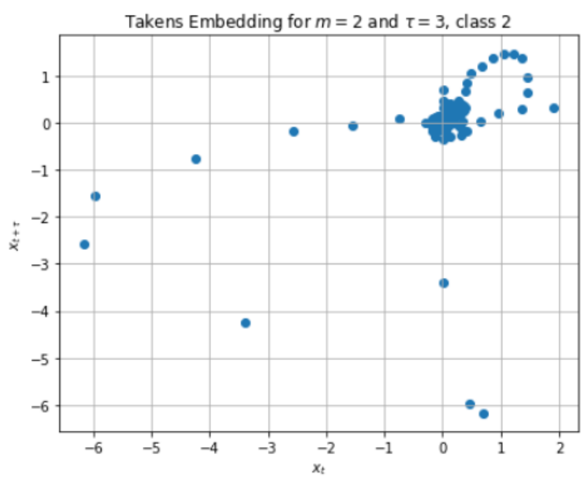

Figure 14. Sample point cloud from Takens's embedding of class 2 for *ECGFiveDays* dataset

5.4. Texas Sharpshooter plot

The Texas Sharpshooter plot was built by using the following formula applied to the obtained train and test accuracy:

$$\text{gain} = \frac{\text{received accuracy}}{\text{accuracy of 1-NN Euclid}}$$

The results are presented in Figure 15. We can see that many cases lie in the false-positive area of the Sharpshooter plot. Data points of that region represent cases where we thought we could improve accuracy, but did not. In some cases, we improved accuracy due to the distinguishing peculiarities that were identified by a model. A successful model and hyperparameters selection also helped to outperform the baselines. Potentially, we can improve the outcome gains for other datasets by working on each one in the future, adjusting embedding and model parameters.

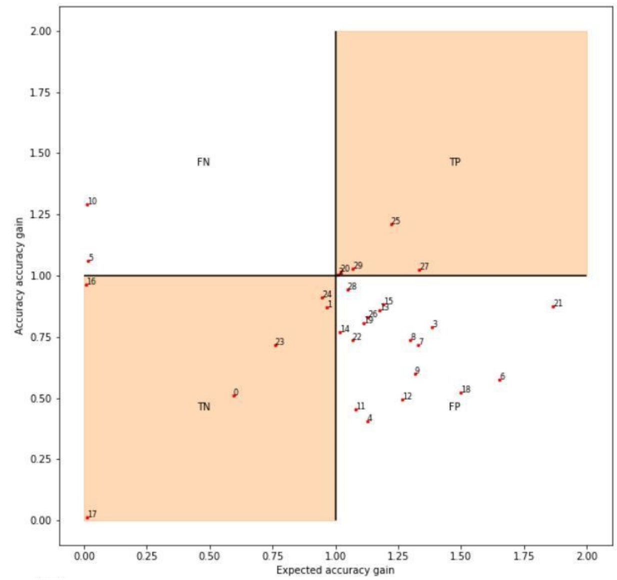

Figure 15. Expected accuracy gain calculated on training data versus actual accuracy gain on testing data, FP: 17, TN: 6, TP: 5, FN: 2

5.5. CD Diagram

Critical difference(CD) diagram is a visualization aimed to rank all algorithms evaluated on the same data collection. The CD diagram is based on the Wilcoxon-Holm method to detect pairwise significance. We used CD diagram to compare the classification algorithms on the UCR datasets collection. It is shown in Figure 16.

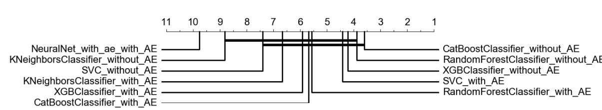

Figure 16. Critical Distance Diagram

A thick horizontal lines on a CD diagram shows a group of classifiers that are not significantly different in terms of accuracy. CatBoost classifier statistically outperforms other algorithms with an average rank about 3.5. Gradient boosting on decision trees algorithms(CatBoost, XGBoost) trained on topological features showed the best perfomance on the average.

6. Conclusion

To sum up, the main purpose of the project that is to study and implement topological features based approach for time series classification and apply proposed pipeline for UCR collection of datasets. The proposed features utilizes constructions like Persistence Diagram, Betti Curves, Persistence Landscape and Persistence Entropy. Those features allow to handle the shape properties of data which are matter in time series classification problem.

Out of the main conclusions of the work is the potential scope of topological features application. Topological features can produce high performance if the point clouds that correspond to each class has distinguishable difference in structure. In initial time domain it can be represented as local or global differences in shape, while the pairwise distance in initial metric space can be very low. Otherwise, topological approach possibly won't bring the performance gain.

Analysing the obtained accuracy scores for UCR dataset and comparing them with the baselines, we can conclude that for some part of the dataset topological features are not applicable due to the structure of time series, or require parameter tuning for obtaining more representative point clouds. As it was mentioned, this parameter tuning is computational-intensive, and speeding up this process can be a direction for further research.

References

Chengyuan Wu, C. A. H. Topological machine learning for multivariate time series. 2019. URL https://arxiv.org/abs/1911.12082.

Dau, H. A., Keogh, E., Kamgar, K., Yeh, C.-C. M., Zhu, Y., Gharghabi, S., Ratanamahatana, C. A., Yanping, Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G., and Hexagon-ML. The ucr time series classification archive, October 2018. https://www.cs.ucr.edu/~eamonn/time_series_data_2018/.

Frédéric Chazal, B. M. An introduction to topological data analysis: fundamental and practical aspects for data scientists. 2017. URL https://arxiv.org/abs/1710.04019.

Goel, A., Pasricha, P., and Mehra, A. Topological data analysis in investment decisions. *Expert Systems with Applications*, 01 2020. doi: 10.1016/j.eswa.2020.113222.

Goodfellow, Ian; Pouget-Abadie, J. M. M. X. B. W.-F. D. O. S. C. A. B. Y. Generative adversarial networks. In *Proceedings of the International Conference on Neural Information Processing Systems (NIPS 2014)*, pp. 2672–2680. 2014. URL https://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

Kennel, M. B., Brown, R., and Abarbanel, H. D. I. Determining embedding dimension for phase-space reconstruction using a geometrical construction. *Phys. Rev. A*, 45:3403–3411, Mar 1992. doi: 10.1103/PhysRevA.45.3403. URL https://link.aps.org/doi/10.1103/PhysRevA.45.3403.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. 2013. URL http://arxiv.org/abs/1312.6114.

Lee M. Seversky, Shelby Davis, M. B. On time-series topological data analysis: New data and opportunities. 2016. URL https://www.cv-foundation.org//openaccess/content_cvpr_2016_workshops/w23/papers/Seversky_On_Time-Series_Topological_CVPR_2016_paper.pdf.

Pereira, C. and Mello, R. Persistent homology for time series and spatial data clustering. *Expert Systems with Applications*, 42, 09 2015. doi: 10.1016/j.eswa.2015.04.010.

Sacan, A., Ozturk, O., Ferhatosmanoglu, H., and Wang, Y. LFM-Pro: a tool for detecting significant local structural sites in proteins‡. *Bioinformatics*, 23(6): 709–716, 01 2007. ISSN 1367-4803. doi: 10.1093/bioinformatics/btl685. URL https://doi.org/10.1093/bioinformatics/btl685.

Takens, F. Detecting strange attractors in turbulence. In Rand, D. and Young, L.-S. (eds.), *Dynamical Systems and Turbulence, Warwick 1980*, pp. 366–381, Berlin, Heidelberg, 1981. Springer Berlin Heidelberg. ISBN 978-3-540-38945-3.

Tan, P.-N., Steinbach, M., and Kumar, V. *Introduction to data mining*. Pearson Education India, 2016.

Umeda, Y. Time series classification via topological data analysis. 2017. URL https://www.jstage.jst.go.jp/article/tjsai/32/3/32_D-G72/_pdf.

Yuhei Umeda, Junji Kaneko, H. K. Topological data analysis and its application to time-series data analysis. 2019. URL https://www.fujitsu.com/global/documents/about/resources/publications/fstj/archives/vol55-2/paper15.pdf.

A. Team member's contributions

Explicitly stated contributions of each team member to the final project.

Abdrakhmanov Ildar (25% of work)

- Topological features extraction
- Models evaluation on UCR datasets
- CD diagram plotting
- Preparing the report
- GitHub repository design

Akhtyamov Timur (25% of work)

- Topological features implementation
- PD extraction implementation
- Models evaluation on UCR datasets
- Preparing Sections 5.4, 6 of report

Mohamed Samir (25% of work)

- Variational Autoencoder implementation
- Parallel the topological features implementation, and the running models
- Evaluation scripts implementation
- Models evaluation on UCR datasets
- Preparing Sections of report

Nikolaeva Anna (25% of work)

- Topological features implementation
- Texas Sharpshooter plot implementation
- Models evaluation on UCR datasets
- Results preparation
- Preparing Sections of report

B. Reproducibility checklist

Answer the questions of following reproducibility checklist. If necessary, you may leave a comment.

1. A ready code was used in this project, e.g. for replication project the code from the corresponding paper was used.

   ☐ Yes.
   ☑ No.
   ☐ Not applicable.

   General comment: If the answer is **yes**, students must explicitly clarify to which extent (e.g. which percentage of your code did you write on your own?) and which code was used.

   Students' comment: The ready code was used only to build a CD Diagram. https://github.com/hfawaz/cd-diagram

2. A clear description of the mathematical setting, algorithm, and/or model is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

3. A link to a downloadable source code, with specification of all dependencies, including external libraries is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: Dependencies and instructions are specified in GitHub repository readme.

4. A complete description of the data collection process, including sample size, is included in the report.

   ☐ Yes.
   ☐ No.
   ☑ Not applicable.

   Students' comment: External ready datasets were used.

5. A link to a downloadable version of the dataset or simulation environment is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

6. An explanation of any data that were excluded, description of any pre-processing step are included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

7. An explanation of how samples were allocated for training, validation and testing is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: train/test parts of all UCR datasets're downloaded directly from http://www.timeseriesclassification.com

8. The range of hyper-parameters considered, method to select the best hyper-parameter configuration, and specification of all hyper-parameters used to generate results are included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

9. The exact number of evaluation runs is included.

   ☐ Yes.
   ☐ No.
   ☑ Not applicable.

   Students' comment: None

10. A description of how experiments have been conducted is included.

    ☑ Yes.
    ☐ No.
    ☐ Not applicable.

    Students' comment: None

11. A clear definition of the specific measure or statistics used to report results is included in the report.

    ☑ Yes.
    ☐ No.
    ☐ Not applicable.

    Students' comment: None

12. Clearly defined error bars are included in the report.

    ☑ Yes.

- ☐ No.
- ☐ Not applicable.

Students' comment: None

13. A description of the computing infrastructure used is included in the report.

- ☑ Yes.
- ☐ No.
- ☐ Not applicable.

Students' comment: None