Protein Classification with Improved Topological Data Analysis

Tamal K. Dey
Department of Computer Science and Engineering, The Ohio State University, Columbus, USA
`http://web.cse.ohio-state.edu/~dey.8/`
dey.8@osu.edu

Sayan Mandal
Department of Computer Science and Engineering, The Ohio State University, Columbus, USA
`http://web.cse.ohio-state.edu/~mandal.25/`
mandal.25@osu.edu

Abstract

Automated annotation and analysis of protein molecules have long been a topic of interest due to immediate applications in medicine and drug design. In this work, we propose a topology based, fast, scalable, and parameter-free technique to generate protein signatures.

We build an initial simplicial complex using information about the protein's constituent atoms, including its radius and existing chemical bonds, to model the hierarchical structure of the molecule. Simplicial collapse is used to construct a filtration which we use to compute persistent homology. This information constitutes our signature for the protein. In addition, we demonstrate that this technique scales well to large proteins. Our method shows sizable time and memory improvements compared to other topology based approaches. We use the signature to train a protein domain classifier. Finally, we compare this classifier against models built from state-of-the-art structure-based protein signatures on standard datasets to achieve a substantial improvement in accuracy.

2012 ACM Subject Classification Applied computing → Life and medical sciences

Keywords and phrases topological data analysis, persistent homology, simplicial collapse, supervised learning, topology based protein feature vector, protein classification

Digital Object Identifier 10.4230/LIPIcs.WABI.2018.6

Supplement Material `http://web.cse.ohio-state.edu/~dey.8/proteinTDA`

Acknowledgements This work has been supported by NSF grants CCF-1318595, CCF-1526513, and CCF-1733798.

1 Introduction

Proteins are by far the most anatomically intricate and functionally sophisticated molecules known. The benchmarking and classification of unannotated proteins have been done by researchers for quite a long time. This effort has direct influence in understanding behavior of unknown proteins or in more advanced tasks as genome sequencing. Since the sheer volume of protein structures is huge, up till the last decade, it had been a cumbersome task for scientists to manually evaluate and classify them. For the last decade, several works aiming at automating the classification of proteins have been developed. The majority of annotation and classification techniques are based on sequence comparisons (for example in BLAST [19], HHblits [2] and [18]) that try to align protein sequences to find homologs or a common ancestor. However, since those methods focus on finding sequence similarity, they are more

© Tamal K. Dey and Sayan Mandal;
licensed under Creative Commons License CC-BY

18th International Workshop on Algorithms in Bioinformatics (WABI 2018).
Editors: Laxmi Parida and Esko Ukkonen; Article No. 6; pp. 6:1–6:13
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

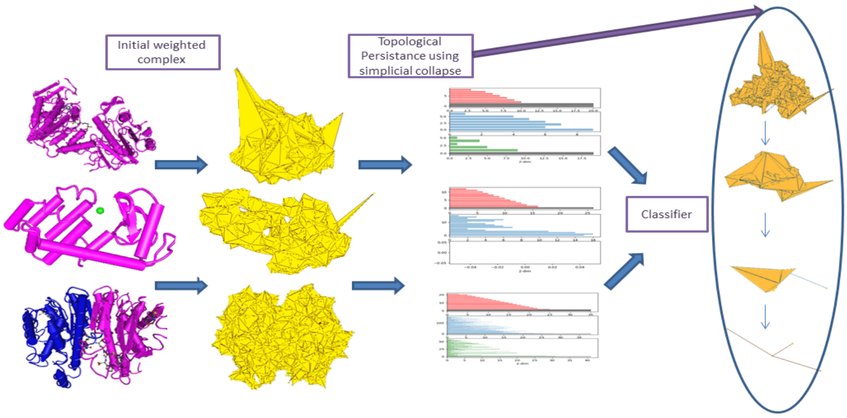

Figure 1 Workflow of our technique.

efficient in finding close homologs. Some domains such as remote homologs are known to have less than 25% sequential similarity and yet have common ancestors and are functionally similar. So, we miss out important information on structural variability while classifying proteins solely based on sequences. Even though, sometimes, homology is established by comparing structural alignment [14], accurate and fast structural classification techniques for the rapidly expanding Protein Data Bank remains a challenge.

Several works on the classification of protein structures exist in the literature. The main intuition behind these works draws upon a heuristic that generates a signature for each protein strand so that structurally close proteins have similar signatures. Essentially, the signature alignment quantifies the similarity between two protein structures. The problem, however, remains with the speed of computing these signatures and the degree of their representative power. We want a fingerprint for the protein that can be computed fast and can tell whether two proteins are dissimilar or even marginally similar.

Some works use vector of frequencies to describe structural features while others take various physical properties into account such as energy, surface area, volume, flexibility/rigidity or use other features from geometric modeling. The "Bag-Of-Word" (BOW) representation to describe an object has been used in computer vision, natural language processing and various other fields. The work by Budowski-Tal [3] have described protein structure using a fragment library in a similar context. Since we use this work for comparison, we shall discuss its details later.

Topological data analysis [10], a newly developed data analysis technique has been shown to give some encouraging results in protein structure analysis. Topological signatures, particularly based on Persistent Homology, enjoy some nice theoretical properties including their robustness and scale invariance. These features are global and more resilient to local perturbations. Moreover, they are invariant to scaling and any isometric transformation of the input. The authors in [23] extract topological fingerprints based on the alignment of atoms and molecules in three dimensional space. Their work shows the impact of persistent homology in the modeling of protein flexibility which is ultimately used in protein B-factor analysis. This work also characterizes the evolution of topology during protein folding and thereby predicts its stability. For this task, the authors have introduced a coarse grain (CG) representation of proteins by considering an amino acid molecule as an atom C_α. This helps them describe the higher level protein structures using the topological fingerprint perfectly. However, since the CG homology may be inconsistent due to ambiguity in choosing the CG particle, we present a similar study on secondary structures using our signature and show that our method does not require such a representation as it is inherently scaling independent.

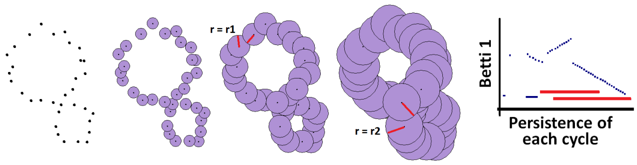

Figure 2 Persistence of a point cloud in $\mathbb{R}^2$ and its corresponding barcode

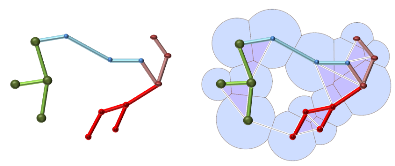

Figure 3 Weighted Alpha complex for protein structure

The authors in [4] have used persistent homology to generate feature vector in the context of machine learning algorithms applied to protein structure explorations. We explore further to improve upon the technique to eliminate its deficiencies. First, the approach in [4] does not differentiate between atoms belonging to different elements. Also, it does not account for the existing chemical bonds between the atoms while building the signature. Most importantly it uses Vietoris Rips(VR) complex to generate the topological features for protein complex which suffers from the well-known problem of scalability. As we will describe later, the VR complex developed in the early 20th century grows rapidly in size even for moderate size protein structures. Current state-of-the-art techniques, which have addressed the problem to some extent, are still very cumbersome and slow especially for structures having about 30,000 atoms on an average. Among the several methods that generate persistence signature from a point cloud, the PHAT toolbox [1] based on several efficient matrix reduction strategies and GUDHI [22] library based on some compression techniques have been popular because of their space and time efficiencies. A recent software called SimBa [8] published last year, has been shown to work faster for large datasets. Yet, for our application, SimBa falls short as we shall see later.

The algorithm that we present here is a fast technique to generate a topological signature for protein structures. We build our signature based on the coordinates of the atoms in $\mathbb{R}^3$ using their radius as weights. Since we also consider existing chemical bonds between the atoms while building the signature, we believe that the hierarchical convoluted structure of protein is captured in our features. Finally, we have developed a new technique to generate persistence that is much quicker and uses less space than even the current state-of-the-art such as SimBa. It helps us generate the signature even for reasonably large protein structures. In sum, in this paper, we focus on three problems: (1) effectively map a protein structure into a suitable complex; (2) develop a technique to generate fast persistent signature from this complex; (3) use this signature to train a machine learning model for classification and compare against other techniques. Our entire method is summarized in figure 1. We also illustrate this method using a supplementary video available at `https://youtu.be/yfcf9UWgdTo`.

2 Methods

We use the theory of topological persistence to generate features for protein structures. These topological features serve as a distinct signature for each protein strand. In this section, we give some background on persistent homology followed by how we construct our signature.

2.1 Persistence signature of point cloud data

We start with a point cloud data in any n-dimensional Euclidean space. These will essentially be the centers of protein atoms in the three dimensional space. However, to illustrate the theory of persistent homology, we consider a toy example of taking a set of points in two dimensions sampled uniformly from a two-hole structure (Fig. 2). We start growing balls around each point, increasing their radius r continually and tracking the behavior of the union of these growing balls. If we start increasing r from zero, we notice that at $r = r_1$ (third from left in Fig 2) both holes are prominent in the union of ball structure. Further increasing r to r_2, leads to filling of the smaller hole (fourth figure from left). This continues till the value of r is large enough for the union of balls to fill the entire structure. During the change in the structure of the union of balls due to increase in radius, the larger of the two holes *'persists'* for a larger range of r compared to the smaller one. Hence features that are more prominent are expected to persist for longer periods of increasing r. This is the basic intuition for topological persistence. The holes in this example are captured by calculating a set of *birth-death* pairs of homology cycle classes that indicate at which value of r the class is born and where it dies. The persistence is visualized in $\mathbb{R}^2$ using horizontal line segments that connect two points whose x-coordinates coincide with the birth and death values of the homology classes. These collection of line segments, as shown in Figure 2, are called barcodes [5]. The length of each line segment corresponds to the persistence of a cycle in the structure. Hence, the short blue line segments correspond to the tiny holes that are formed intermittently as the radius increases. The two long red line segments correspond to the two holes in the structure, the largest being the bigger hole. For computational purposes, the growing sequence of the union of balls is converted to a growing sequence of triangulations, simplicial complexes in general, called a *filtration.* In some cases, some cycles called the *'essential cycles'* persists till the end of the filtration.

The rank of the persistent homology group called the persistent Betti numbers capture the number of persistent features. For n-dimensional homology group, we denote this number as β_n. This means β_0 counts the number of connected components that arise in the filtration. Similarly, β_1 counts the number of *circular* holes being born as we proceed through the filtration. It is due to this fact that all the folds in the tertiary structure, as well as the helix and strands in the secondary structure of proteins, are recorded in our signature.

With the above technique, difficulties are faced as r increases. An average protein in a database such as CATH [20] has 20,0000~30,000 atoms, thus creating a point cloud of the same size in $\mathbb{R}^3$. Furthermore, the initial complex including 3-simplices (or tetrahedra) becomes quite large. On an average, this complex size grows to (50~100)x10^4 simplices of dimension upto 4 and becomes quite difficult to process. Building a filtration using this growing sequence of balls is thus not scalable. We attack the problem with two strategies: (1) we only consider simplices on the boundary of the entire simplicial complex in our algorithm and (2) compute a new filtration technique that is based on collapsing simplices rather than growing their numbers by addition.

Topological persistence

Traditionally, given a point cloud, its persistence signature is calculated by building a filtration over a simplicial complex called *Vietoris-Rips*(VR). This technique is also used in [4] which takes the 3D position of the centers of the atoms as points in the point cloud. Given a parameter α, we can define VR complex over a point cloud P as: $\mathcal{VR}^{\alpha}(\mathbf{P}) = \{\sigma \mid \mathbf{d}(\mathbf{p}, \mathbf{q}) < \alpha \;\forall\; \mathbf{p}, \mathbf{q} \in \sigma\}$.

As the value of α increases, more edges and higher order simplices are introduced, and a *filtration* is obtained. Finally, the persistence of this *filtration* is computed. For a better representation of protein molecules, we take into account the radius of different atoms as weight of the points. So, we replace each point $p \in P$ with a tuple $\hat{p} = (p, r_p)$ where r_p is the radius of the atom represented by p. For the resulting weighted point cloud $\hat{P} = \{(p, r_p)\}$, we consider the weighted VR complex: $\mathcal{VR}^{\alpha}(\hat{\mathbf{P}}) = \{\sigma \mid \mathbf{d}(\mathbf{p}, \mathbf{q}) < \alpha(\mathbf{r_p} + \mathbf{r_q}) \;\forall\; \mathbf{p}, \mathbf{q} \in \sigma\}$.

The VR complex is easy to implement, but its size can become a hindrance for an even a moderate size protein molecule. Thus, instead of a VR complex, we use the (weighted) alpha complex that is sparser and has been used to model molecules in earlier works [11].

Alpha complex AC($\boldsymbol{\alpha}$): For a given value of α, a simplex $\sigma \in AC(\alpha)$ if:

- The circumball of σ is empty and has radius $< \alpha$, *or*
- σ is a face of some other higher dimensional simplex in AC(α).

Weighted Alpha Complex $\boldsymbol{WAC_{\hat{P}}(\alpha)}$: Let $B_k(\hat{p})$ be a k-dimensional closed ball with center p, and weight r_p. It is orthogonal or sub-orthogonal to a weighted point (p',$r_{p'}$) iff $\mathbf{||p-p'||^2 = r_p^2 + r_{p'}^2}$ **or** $\mathbf{||p-p'||^2 < r_p^2 + r_{p'}^2}$ respectively.

An *orthoball* of a k-simplex $\sigma = \{\hat{p}_0, \ldots, \hat{p}_k\}$ is a k-dimensional ball that is orthogonal to every vertex p_i. A simplex is in the weighted alpha complex $WAC_{\hat{P}}(\alpha)$ iff its orthoball has radius less than α and is suborthogonal to all other weighted points in $\hat{P}$.

2.2 Collapse-induced persistent homology from point clouds

The following procedure computes a topological signature for a weighted point cloud $\hat{P} = \{p, r_p\}$ using subsamples and subsequent collapses:

1. Compute a weighted alpha complex $\mathbb{C}^0$ on the point set $\hat{P} = \{p, r_p\}$ using the algorithm described in [22]. Let V^0 be the vertex set of $\mathbb{C}^0$.
2. Compute a sequence of subsamples $V^0 \supset V^1 \supset ... \supset V^k$ of the initial vertex set V^0 based on the Morton Ordering as discussed later. (For every V^i, we remove every n^{th} point in the Morton Ordering from V^i to form V^{i+1}. We choose 'n' based on the number of initial points).
3. This sequence of subsets of V^i allows us to define a simplicial map between any two adjacent subsets V^i and V^{i+1}: $f^i(p) = \begin{cases} p & \text{if } p \in V^{i+1} \\ \underset{v \in V^{i+1}}{\text{argmin}}\ d(p,v) & \text{otherwise} \end{cases}$
4. This vertex map $f^i : V^i \to V^{i+1}$ in turn generates a sequence of collapsed complexes: $\mathbb{C}^0, \mathbb{C}^1, ..., \mathbb{C}^n$. Each vertex map induces a simplicial map $f^i : \mathbb{C}^{i-1} \to \mathbb{C}^i$ that associates simplices in $\mathbb{C}^{i-1}$ to simplices in $\mathbb{C}^i$(see Figure 4)
5. Compute the persistence for the simplicial maps in the sequence $\mathbb{C}^0 \xrightarrow{f_1} \mathbb{C}^1 \xrightarrow{f_2} ... \xrightarrow{f_k} \mathbb{C}^k$ to generate the topological signature of the point set $\hat{P}$.

In step 1 of the procedure, weighted points alone lead to disconnected weighted atoms in $\mathbb{C}^0$ rather than capturing the actual protein structure. To sidestep this difficulty, we increase the weights of these points based on the existence of covalent or ionic bonds in the structure. That is, if there exists a chemical bond between two atoms (which we get from the input .pdb file), we scale-up the weight of each point so that they are connected in the weighted alpha complex $WAC_{\hat{P}}(\alpha)$ (see Fig. 3). We determine a global multiplying factor $\rho \geq 1$ for this purpose. As mentioned earlier, we take the boundary of this weighted complex which forms our initial simplicial complex $\mathbb{C}^0$.

In step 2, in order to generate the sequence of subsamples, we pick vertices uniformly from the simplicial complex to be collapsed to their respective nearest neighbors. To choose a subsample that respects local density, we use a space curve generation technique called Morton Ordering [15]. The Morton curve generates a total ordering on the point set V^0.

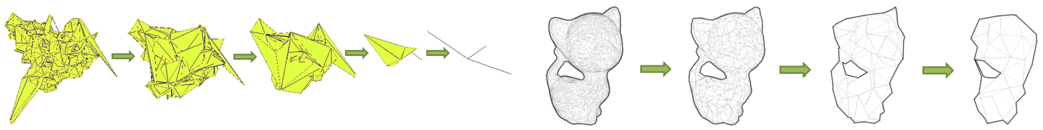

Figure 4 (a) Collapse of weighted alpha complex generated from protein structure via simplicial map. (b) Same algorithm applied to a kitten model in $\mathbb{R}^3$.

This ordering is explicitly defined by the Morton Ordering map $M : \mathbb{Z}^N \mapsto \mathbb{Z}$ given by:

$$\mathbf{M(p)} = \bigvee_{\mathbf{b=0}}^{\mathbf{B}} \bigvee_{\mathbf{i=0}}^{\mathbf{N}} \mathbf{x_2^{i,b}} \ll \mathbf{N(b+1)} - \mathbf{(i+1)},$$

where $x_2^{i,b}$: b^{th} bit value of the binary representation of the i^{th} component of x.
This map merely interleave bits of the different components of p. Application of M to V^0 yields a total ordering on our initial point set. To generate a new subset $V^1 \subset V^0$, we simply choose a value n such that $1 < n \leq \|V^0\|$. Then, V^{i+1} is taken as:

$$\mathbf{V^{i+1}} = \{\mathbf{x_j} \mid \mathbf{x_j} \in \mathbf{V^i}, \mathbf{j} \not\equiv \mathbf{0} \bmod \mathbf{n}\},$$

where x_j is the j^{th} vertex in the Morton Ordering of V^i. We choose $n = 12$ as it has procured good results for the datasets we experimented on (having 20,000~30,000 atoms on an average). Following this approach, the process can be repeated to create a sequence of subsets $V^0 \supset V^1 \supset ... \supset V^n$, $\|V^n\| \leq k$ as done in step 2 of our procedure above.

Finally, as described in step 3, instead of constructing the filtration by increasing the value of α, we perform a series of successive collapses starting with the initial simplicial complex. This leads to a sequence of complexes that decreases in size instead of growing as we proceed forward. Effectively, it generates a sequence called *tower* of simplicial complexes where successive complexes are connected by *simplicial maps*. These maps which are the counterpart of continuous maps for the combinatorial setting extend maps on vertices (vertex maps) to simplices (see [16] for details). In our case, collapses of vertices generate these simplicial maps between a simplicial complex in the tower to the one it is collapsed to. Persistence for towers under simplicial maps can be computed by the algorithm introduced in [7]. We use the package called Simpers that the authors have reported for the same.

To summarize, the algorithm generates an initial weighted alpha complex. It then proceeds by recursively choosing vertices based on Morton Ordering to be collapsed to their nearest neighbors resulting in vertex maps. These vertex maps are then extended to higher order simplices (such as triangles and tetrahedra) using the simplicial map. Finally given the simplicial map, we generate the persistence and get the barcodes for the zero and one dimensional homology groups.

2.3 Feature vector generation

We discuss how we generate a feature vector given a protein structure. We take protein data bank (*.pdb) files as input to extract protein structures. It contains the coordinates of every atom, their name, chemical bond with neighboring atoms and other meta-data such as helix, sheet and peptide residue information. We introduce a weighted point for each atom in the protein where the point is the center of the atom and its weight is the specified radius. For instance, for a Nitrogen atom in the amino acid, we assign a weight equal to its covalent

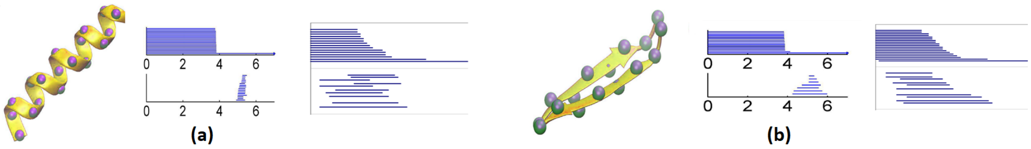

Figure 5 (a) Left: Alpha helix from PCB 1C26 , Middle: Barcode of [23], Right: Our Barcode, (b) Left: Beta sheet from PCB 2JOX, Middle: Barcode of [23], Right: Our Barcode. Each segment of the barcodes shows $\beta 0$(top) and $\beta 1$(bottom).

radius of 71(pm). On this weighted point cloud $\hat{p} = (p, r_p)$, if two atoms $\hat{p}$ and $\hat{q}$ are involved in a chemical bond, we increase their weights so that p and q get connected in the alpha complex. We compute the persistence by generating the initial alpha complex and undergoing a series of collapses as described in the previous section. For computational efficiency, we only consider the barcodes in zero and one dimensional homology groups. Note that some of the barcodes can have death time equal to infinity indicating an essential feature. For finite barcodes, shorter lengths ($death - birth$) indicate noise. Elimination of these intermittent features serves some interesting purpose as we will see in section 3. To find relatively long barcodes, we sort them in descending order of their lengths. Let $\{l_1, l_2, ..., l_k\}$ be this sorted sequence. Consider the sequence $\{l'_1, l'_2, ..., l'_{k-1}\}$ where $l'_i = l_{i+1} - l_i$ and let l'_m be a maximal element for $1 \leq m \leq k-1$. All barcodes with the lengths $[l_1..l_m]$ form part of the feature vector. Essentially we remove all barcodes whose lengths are shorter than the largest gap between two consecutive barcodes when sorted according to their lengths. A similar technique used in [13] has shown improved results in image segmentation over other heuristics and parameterizations. Since the feature vector needs to be of a fixed length for feeding into a classifier, we compute the index m of l_m over all protein structures and take an average. The feature vector also includes the number of *essential* zero and one dimensional cycles. Therefore, we have a feature vector of length $2 \times m + 2 : \{l^0_1, l^0_2, ...l^0_m, l^1_1, l^1_2, ...l^1_m, c_{\beta_0}, c_{\beta_1}\}$. Here l^0_i and l^1_i are the lengths of zero and one dimensional homology cycles respectively whereas c_{β_i} are the total number of essential cycles in i-dimensional homology.

3 Experiments and results

We perform several experiments to establish the utility of the generated topological signature. First, we show how our feature vector captures various connections in the single strands of secondary structures and compare them against the signatures obtained in [23]. Then we investigate if there is a correlation between the count of such secondary structures and our feature vector. Next, we describe the topological feature vector obtained from two macromolecular proteins structures. We also compare the size and time needed by our algorithm (software) over the other commonly used persistence software (as in [4]). Lastly, we show the effectiveness of our approach in classifying protein structures using machine learning models.

Topological description of alpha helix and beta sheet

It is known that barcodes can explain the structure of an alpha helix and a beta sheet [23]. The authors in [23] use a coarse-grain(CG) representation of the protein by replacing each amino acid molecule with a single atom. This representation removes the short barcodes corresponding to the edges and cycles of the chemical bonds inside the amino acid molecule.

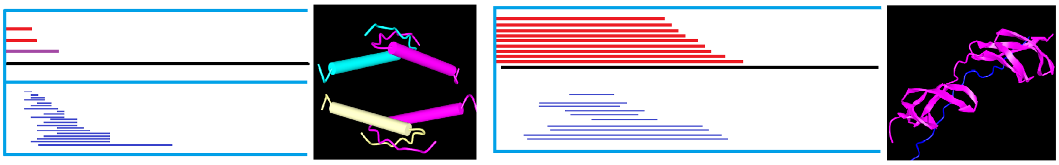

Figure 6 Barcode and Ribbon diagram of (Left): PDB: 1c26. (Right): PDB: 1o9a. Diagram courtesy NCBI [17].

We do not need this CG representation as our procedure can implicitly determine a threshold l_m and therefore delete all barcodes of length shorter than the largest gap between two consecutive barcodes (as described in section 2.3). So, we get a barcode that describes the essential features of the secondary structures without including noise or short lived cycles from the amino acids. For a fair comparison, we compute our barcodes on the same alpha helix residue as in [23] with 19 residues extracted from the protein strand having PDB ID 1C26 (see figure 5). Analogous to the barcode of [23] (as shown in the middle diagram of figure 5a), we have 19 bars in the zero-dimensional homology for the alpha helix representing the nineteen initial residues. These components die as edges are introduced in the weighted alpha complex which gets them connected. For one-dimensional homology, an initial ring with 4 residues is formed followed by additional rings resulting from the growing connections in each amino acid. These cycles eventually die by the collapse operations in our algorithm.

The same process is followed for beta sheets after we extract two parallel beta sheet strands from the protein structure with PDB ID 2JOX. The zero-dimensional homology cycles are killed when individual disconnected amino acid residues belonging to the same beta sheet strand are connected by edges, as represented in the top 17 barcodes (leftmost figure of 5b). However, other than these barcodes and the longest bar corresponding to the *essential cycle*, there is one bar in the zero-dimensional homology which is longer than the top 17 bars. This bar represents the component which is killed by joining the closest adjacent amino acid molecules from the two parallel beta strands. The one dimensional homology bars are formed as more adjacent amino acid molecules are connected and killed once the collapse operation starts. Note that the two barcodes shown in figure 5 comparing our work with [23] are not to scale. This is because, in contrast to [23], the barcodes in our figure are not plotted against Euclidean distance rather the step at which each insertion and collapse operation occurs.

A caveat

Our aim is to compute signatures that capture discriminating structural information useful for classifying proteins. Even though we can use our signature to describe secondary structures, we do not want our signature to be directly correlated to the *number of* alpha helix or beta sheet as it would mean they are redundant. We generate a 2 × 12 matrix where each cell contains the correlation value between beta-sheet(top row) and alpha-helix(bottom row) with each individual component in the feature vector: $\{l_1^0, l_2^0, ...l_m^0, l_1^1, l_2^1, ...l_m^1, c_{\beta 0}, c_{\beta 1}\}$. We use proteins in the PCB00020 record of the PCBC database to compute this matrix and depict it by a heatmap (Fig 7). Essentially, we first generate two vectors v_α and v_β of the number of alpha helices and beta sheets respectively in each protein over all entries in the database. Similarly, we produce a vector for each value in the feature vector: $\{v_{l_1^0}, ..., v_{c_{\beta 1}}\}$. Now we populate the matrix by calculating the correlation between each of these individual vectors with v_α and v_β. For example, row 1 and column 1 of the matrix contain the correlation value between the vectors $v_{l_1^0}$ and v_β. The heat map color ranges from blue for

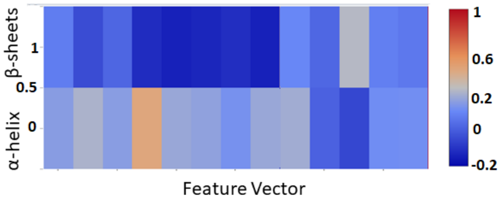

Figure 7 Heatmap correlating secondary structure against our feature vector. Each column in the heatmap is the feature vector.

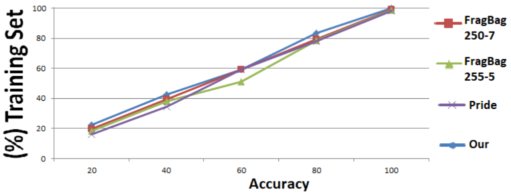

Figure 8 Plot showing accuracy against varying training data size. 100(%) indicates the entire training and test data.

zero correlation to dark-red for complete correlation. As we can see from the figure, almost all matrix entries have a blue tinge indicating low correlation. This shows that our feature vector is non-redundant over the frequency of secondary structures.

Topological Description of macromolecular structures

In the previous section, we use our signature to describe the secondary structures and compare it with the work in [23]. In this section, we further show how our signature works by describing two macromolecular protein structures that are built on multiple secondary structures. We start by describing the tetrametric protein: 1C26. The ribbon diagram and associated barcode after noise removal is given in figure 6 . It essentially contains four monomers, associated pairwise to form two dimers. These two dimers, in turn, join across a distinct parallel helix-helix interface to form the tetramer. When we build the filtration on this protein structure, two monomers on opposite sides are killed first by connecting to their adjacent monomers to form two distinct dimers. This is evident as there are two short bars in the zero dimensional barcode (Fig. 6 right: shown in red). We now have two dimers, one of which is killed when it joins with the other to form a third slightly longer non-essential barcode (shown in purple). The second dimer lives on as the tetramer and forms an essential barcode (shown in black). Next, if we look into the one dimensional homology (shown as blue lines), we notice that the most notable feature for the protein is the tetramer structure which contains a large loop when the two dimers are connected. This is evident in our 1D-barcode as there is a distinct long bar representing the large one dimensional cycle. Note that the birth time of this cycle in 1D corresponds with the death time of the non-essential dimer in 0D.

Next, we consider the protein structure 1O9A. The structure contains several antiparallel beta-strands and is an example of a tandem beta-zipper. As we can see from the ribbon diagram in Fig. 6, there are six beta sheets on one side and five on another, connected together to form a fibronectin. This is evident as there are ten non essential and one essential bar in the zero dimensional homology owing to the six beta sheets on one side and five on the other. Each component is killed as the beta sheets join with another as the filtration proceeds. Note that the last connected component after joining all beta sheets forms an essential bar. Moreover, since there is no distinct cycle in the structure, we do not get any distinct long bar in the one dimensional homology. The presence of multiple one dimensional bars of similar size are probably due to the antiparallel beta-strands on either side which form a ring once joined. Thus, we can see that using the same signature generation method, we can describe secondary structures (as in the previous section) as well as macromolecular proteins without any change in the parameter. It is therefore evident that our signature is intrinsic and scale independent.

Table 1 Time comparison of our algorithm against SimBa [8] and VR complex.

		Size			Time (in sec)		
Data	Dim	VR	SimBa	Our	VR	SimBa	Our
CATH	3	–	1422	443	–	1.75	0.35
Soneko	3	324802	10188	576	32	6.77	2.05
Surv-l	150	–	3.1×10^6	1.09×10^6	–	5.08×10^3	884
PrCl-I	25	–	10.2×10^6	0.22×10^6	–	585	141.3

Table 2 Accuracy comparison with FragBag and Cang.

	SVM			KNN		
	FB	Cang	Our	FB	Cang	Our
Class	91.08	89.07	92.36	86.01	86.40	86.39
Architecture	90.26	91.11	92.20	88.17	87.47	89.11
Topology	92.19	94.87	96.71	91.54	94.02	96.20
Homology	93.33	94.06	94.17	90.28	91.11	93.30

Time and space comparison with VR-complex and SimBa

The method in [4] uses persistent homology as feature vectors for machine learning. However, as mentioned earlier, the use of Veitoris-Rips (VR) complex leads to a size blow up that not only increases runtime but also in most cases, causes the operating system to abort the process due to space requirements. Results in [4] procure good results as the datasets are of moderate size, but the same could not be reported for larger and real life protein structures. In table 1, we show a size and time comparison of our approach with the original feature generation technique used in [4]. We also tabulate the size and time to generate the same feature vector in [4] using a state-of-the-art persistence algorithm called SimBa [8]. Table 2 contains a mix of protein databases and other higher dimensional datasets. As we see in the table, our algorithm is faster even when the features in [4] are generated with SimBa.

3.1 Supervised learning based classification models

Classification model. For the purpose of protein classification, we train two classifiers: an SVM model and a k-nn model on some protein databases. Once the model is trained, we test it to find accuracy, precision, and recall. The reason behind choosing Support Vector Machine and k-nn based supervised learning technique over other sophisticated and state-of-the-art classifiers is their basic nature. Results obtained from basic learning techniques prove the effectiveness of the feature vectors rather than that of the classifier. We can further improve the classification accuracy for proteins using some advanced supervised learning or Neural Network based classifiers using our proposed features.

Benchmark techniques. In order to test the effectiveness of our protein signature, we need to compare it against some of the state-of-the-art protein structure classification techniques. We generate feature vectors through these techniques and train and test the same classification models as before. The first technique, known as PRIDE [9], classifies proteins at the Homology level in the CATH database. It represents protein structures by distance distributions and compares two sets of distributions via contingency table analysis. The more recent work by Budowski-Tal et al. [3], which has achieved significant improvement in protein classification is used as our second benchmark technique. Their work, known as FragBag mimics the bag of word representation used in natural language processing and computer vision. They maintain protein fragments as a benchmark. Given a large protein strand, they generate a vector which is essentially a count of the number of times each fragment approximates a segment in this strand. This vector now acts as a signature for the protein structure and that is what forms the basis for their feature vector which we use to train and test our classifier. The protein fragment benchmark is available from the library [12]. We choose 250 protein fragments of length 7. The third work that we test against is the topological approach to generate a protein fingerprint [4]. However, as we saw earlier, it is not possible to generate

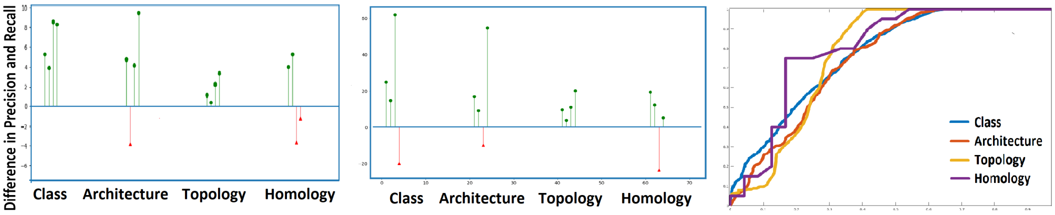

Figure 9 Left:a) Difference in precision and recall from FragBag. Middle: b) Difference in precision and recall from [4]. Right: c) ROC curve for SVM classification of our algorithm.

all the protein signatures using the original algorithm used by the authors. Therefore, we replace the Vietoris-Rips filtration by the state-of-the-art SimBa and generate feature vectors the same way as mentioned in their paper.

Database. The database that we use is called Protein Classification Benchmark Collection (PCBC) [21]. It has 20 classification tasks, each derived from either the SCOP or CATH databases, each containing about 12000 protien structures. The classification experiment involves the group of the domain as positive set, and the positive test set is the sub-domain group. The negative set includes the rest of the database outside the superfamily divided into a negative test or negative training set. The result for some of the classification tasks for the database is given in Table 3. As evident from the table, the accuracy obtained by using our signature has a considerable improvement over the state of the art techniques. The only classification task in which our algorithm under-performs is with the protein domain CATH95_Class_5fold_Filtered (fourth row of table 3). The class domain is randomly sub-divided into 5 subclasses in this task. Since the class is divided randomly into subclasses, we believe some proteins belonging to different sub-classes have generated a similar initial complex resulting in a similar filtration and ultimately a decrease in performance.

The PCBC dataset, even though suitable for learning algorithms, suffers from being skewed as the number of negative instances in any classification is much larger than the number of positive instances, leading to probable incorrect classifications. Therefore, we test on one of the most popular protein databases known as CATH [6]. The CATH database contains proteins divided into different domains (C: class; A: architecture; T: topology; H: homologous superfamily). For each domain, we get protein structures and their labels in accordance with the sub-domain they belong to. For any classification task, we randomly choose positive instances from one sub-domain and the same number of negative instances sampled equally from the other sub-domains. Each such task, on average has 400 protein structures containing approximately 30,000 atoms each. We then divide this into 80%-training and 20%-test set. The result of classification on the CATH database averaged over several such randomly chosen sub-domains as positive classes, are illustrated in table 2. We see yet again that for each case, there is an improvement of about 3-4% over the benchmark techniques.

3.1.1 Classification result

We have listed our main results in tables 2 and 3 showing the improvement in accuracy using our method over the state-of-the-art techniques of FragBag, PRIDE and the preceding work on topology by Cang et al. [4]. We provide further evidence of the efficiency of our algorithm by comparing the precision and recall in figures 9a and b. In these plots, we show

Table 3 Classification accuracy for different techniques on Protein dataset. SC: SCOP95, CA: CATH95, Sf: Superfamily, Fm: Family, F: Filtered, T: Topology, H: Homology, C: Class, 5f: 5fold, A: Architecture, Si: Similarity.

	SVM				k-NN			
	Pride	Fragbag	Cang	Our	Pride	Fragbag	Cang	Our
SC_Sf_Fm_F	90.09	93.01	93.39	95.24	89.58	87.31	89.83	91.66
CA_T_5f	94.23	92.97	94.87	99.53	90.96	91.16	94.57	97.87
CA_T_H_F	90.15	89.89	95.06	98.80	84.98	81.11	86.65	95.51
CA_C_5f_F	85.09	84.76	80.98	82.36	80.18	84.74	83.83	78.81
CA_H_Si_F	98.60	95.89	98.24	99.05	95.45	91.11	79.469	97.56
CA_A_T_F	87.56	91.58	74.58	90.95	67.47	89.00	68.90	87.00

the difference between the precision and recall obtained using our algorithm against that of FragBag(9a) and Cang(9b). A green bar indicates that our algorithm performed better and the difference is positive while a red bar suggests the opposite. This experiment is done on the CATH database and the figure shows the precision and recall for each domain: class(C), architecture(A), topology(T) and homology(H). Notice that, since the classification is binary, we get two precision and two recall for every class in each domain. Thus, there are four bars for each of C,A,T,H. Yet again, other than a few marginal cases, our algorithm largely performs better. Finally, we calculate the ROC curve using SVM on a subset of the CATH dataset, the result of which is shown in figure 9c. The ROC curve is a plot of the true positive rate against false positive rate obtained by changing the input size and parameter. This means that the further the lines are away from the diagonal, the better is the classifier.

For the positive test cases, we investigate further the trend of the output. We try to see the correlation of accuracy with the change in training set size. We therefore change the training and test set sizes by taking a fraction of the entire dataset and trace the accuracy in each case. This is done over all the test cases shown in Table 3 and the average is shown in Fig 8. We have plotted the output of our algorithm in blue with two instances of FragBag with (fragment, library) sizes (5,225) and (7,250) in red and green respectively. In addition, we have plotted the output of PRIDE as well. Ideally, the accuracy should decrease uniformly with a decrease in training set size and we should get a straight line across the diagonal. In this case, all the trendlines are almost close to the diagonal and hence we can say that they are correlated. Moreover, we observe that even as the training data size decreases, the accuracy of our algorithm remains better or comparable to the other algorithms. This indicates that topological features work better with a lower number of samples as well.

4 Conclusion

We present a practical topological technique to generate signatures for protein molecules that can be used as feature vectors for its classification. Since we investigated the descriptive power of our signature, we believe it can be used for other purposes such as protein energy computation, or finding protein B-factor. We believe that this signature can be extended to other biomolecular data such as DNA or enzymes.

References

1 Ulrich Bauer, Michael Kerber, Jan Reininghaus, and Hubert Wagner. Phat - persistent homology algorithms toolbox. *J. Symb. Comput.*, 78(C):76–90, 2017.

2 Juliana Bernardes, Gerson Zaverucha, Catherine Vaquero, Alessandra Carbone, and Levitt Michael. Improvement in protein domain identification is reached by breaking consensus,

with the agreement of many profiles and domain co-occurrence. *PLoS Computational Biology*, 12, 07 2016.

3 Inbal Budowski-Tal, Yuval Nov, and Rachel Kolodny. Fragbag, an accurate representation of protein structure, retrieves structural neighbors from the entire pdb quickly and accurately. *PNAS*, 107(8):3481–3486, February 2010.

4 Zixuan Cang, Lin Mu, Kedi Wu, Kristopher Opron, Kelin Xia, and Guo-Wei Wei. A topological approach for protein classification. In *Computational and Mathematical Biophysics*. MBMB, Nov 2015.

5 Gunnar Carlsson, Afra Zomorodian, Anne Collins, and Leonidas Guibas. Persistence barcodes for shapes. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, SGP '04, pages 124–135. ACM, 2004.

6 Natalie Dawson, Tony E Lewis, Sayoni Das, Jonathan Lees, David Lee, Paul Ashford, Christine Orengo, and Ian Sillitoe. Cath: An expanded resource to predict protein function through structure and sequence. *Nucleic Acids Research*, 45, 11 2016.

7 Tamal K. Dey, Fengtao Fan, and Yusu Wang. Computing topological persistence for simplicial maps. *Symposium on Computational Geometry*, pages 345–354, june 2014.

8 Tamal K. Dey, Dayu Shi, and Yusu Wang. Simba: An efficient tool for approximating rips-filtration persistence via simplicial batch-collapse. In *ESA*, volume 57 of *LIPIcs*, 2016.

9 Zoltán Gáspári, Kristian Vlahovicek, and Sándor Pongor. Efficient recognition of folds in protein 3d structures by the improved pride algorithm. *Bioinformatics*, 21(15), 2005.

10 Edelsbrunner Herbert and John Harer. *Computational topology: an introduction*. American Mathematical Society, 2010.

11 Liang J, Edelsbrunner H, Fu P, Sudhakar PV, and Subramaniam S. Analytical shape computation of macromolecules: Ii. molecular area and volume through alpha shape. In *Proteins*, volume 33, pages 18–29, 1998.

12 Rachel Kolodny, Patrice Koehl, Leonidas Guibas, and Michael Levitt. Small libraries of protein fragments model native protein structures accurately. *JMB*, 323, 2002.

13 Vitaliy Kurlin. A fast persistence-based segmentation of noisy 2D clouds with provable guarantees. *Pattern Recognition Letters*, 83:3–12, 2015.

14 Holm Liisa and Rosenström Päivi. Dali server: conservation mapping in 3d. *Nucleic Acids Research*, 38:W545–W549, 2010. `doi:10.1137/070711669`.

15 G. M. Morton. A computer oriented geodetic data base; and a new technique in file sequencing. *International Business Machines Co.*, 1966.

16 J. R. Munkres. *Elements of Algebraic Topology*, chapter 1. CRC Press, 1 edition, 1984.

17 USA National Institutes of Health, 1988. URL: `https://www.ncbi.nlm.nih.gov/`.

18 M Remmert, A Biegert, and Söding J. Hauser A. Hhblits: lightning-fast iterative protein sequence searching by hmm-hmm alignment. *Nature Methods*, 9, Dec 2011.

19 Altschul S.F., Gish W., Miller W., Myers E.W., and Lipman D.J.n. Basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410, 1990.

20 Ian Sillitoe, Tony E Lewis, and et al. Cath: Comprehensive structural and functional annotations for genome sequences. *Nucleic Acids Research*, 43, 01 2015.

21 Paolo Sonego, Mircea Pacurar, Somdutta Dhir, Attila Kertesz-Farkas, András Kocsor, Zoltán Gáspári, Jack A M Leunissen, and Sándor Pongor. A protein classification benchmark collection for machine learning. *Nucleic acids research*, 35:D232–6, 02 2007.

22 The GUDHI Project. *GUDHI User and Reference Manual*. GUDHI Editorial Board, 2015. URL: `http://gudhi.gforge.inria.fr/doc/latest/`.

23 Kelin Xia and Guo-Wei Wei. Persistent homology analysis of protein structure, flexibility and folding. *IJNMBE*, 30(8):814–844, 2014. URL: `doi:10.1002/cnm.2655`.