A new method for protein characterization and classification using geometrical features for 3D face analysis: An example of tubulin structures

# A new method for protein characterization and classification using geometrical features for 3D face analysis: an example of tubulin structures

Luca Di Grazia [1], Maral Aminpour [2,3] Enrico Vezzetti [1], Vahid Rezania [4], Federica Marcolin [1], and Jack Adam Tuszynski [1,2,3]*

[1] Politecnico di Torino, corso Duca degli Abruzzi 24, 10129 Torino, Italy;
[2] Department of Physics, University of Alberta, Edmonton, Alberta, Canada
[3] Department of Oncology, University of Alberta, Edmonton, Canada
[4] Department of Physical Sciences, MacEwan University, Edmonton, Alberta, Canada
* Correspondence: jacek.tuszynski@polito.it;

**Short title/running title:** protein characterization and classification using geometrical features for 3D face analysis

**Abstract:** This paper reports on the results of research aimed to translate biometric 3D face recognition concepts and algorithms into the field of protein biophysics in order to precisely and rapidly classify morphological features of protein surfaces. Both human faces and protein surfaces are free-forms and some descriptors used in differential geometry can be used to describe them applying the principles of feature extraction developed for computer vision and pattern recognition. The first part of this study focused on building the protein dataset using a simulation tool and performing feature extraction using novel geometrical descriptors. The second part tested the method on two examples, first involved a classification of tubulin isotypes and the second compared tubulin with the FtsZ protein, which is its bacterial analogue. An additional test involved several unrelated proteins. Different classification methodologies have been used: a classic approach with a Support Vector Machine (SVM) classifier and an unsupervised learning with a k-means approach. The best result was obtained with SVM and the radial basis function (RBF) kernel. The results are significant and competitive with the state-of-the-art protein classification methods. This leads to a new methodological direction in protein structure analysis.

**Keywords:** 3D Face Analysis; Protein Classification; Tubulin; SVM; Geometrical Descriptors; Differential Geometry; Machine Learning

## 1. Introduction

The structure of a protein is an important indicator of its potential biological functions, especially its surface, which is exposed to the solvent and participates in interactions with other proteins and ligands. In a recently published work [1] it was shown how to capture fingerprints of a protein using deep learning methodology and a strong correlation was demonstrated between the structure of a protein and its biological behavior. Another work [2] showed the relevant role of protein-protein interactions using local structural features. In this latter paper geometrical features were found to be interesting in this context.

The first step in the process of classifying proteins is to acquire a realistic (usually experimental) 3D dataset regarding a protein's structure. X-ray crystallography has made the largest and most important contribution to our understanding of protein structure. Nuclear Magnetic Resonance (NMR) and cryogenic electron microscopy (cryo-EM) are other methods by which to determine the protein structure [3] but they have various limitations. As an alternative to crystallographic structure determination, a computational method can be used to generate its prediction using a three-dimensional model [4]. However, proteins are non-static molecular structures, thus a crystallography-generated image is only a snapshot in time of a protein structure and not a fully realistic representation of all protein states, which can be quite dynamic. Therefore, molecular dynamics (MD) is a useful computational tool that can be used to produce atomic coordinate trajectories in order to provide a sampling of structural representations of a given protein. The method we propose in this paper is agnostic to the origin of the data, which in the case of proteins can either be obtained from experiments such as cryo-EM or synthetically generated from computational approaches such as MD. The key aspect is to have an atomistic model of the objects studied [3], which serves as the starting point for feature extraction based on the protein surface.  Such a model provides a high-resolution representation of the object of interest, which is later on processed and characterized by a manageable number of parameters.

A protein can have different equilibrium conformational states that depend on ambient conditions. Moreover, some proteins are expressed by several genes leading to different isotypes with a high degree of structural similarity making accurate comparison important, so a dataset with significant number of different frames is important in order to have a statistically significant and valid test set. The most difficult task would be to distinguish between very closely related proteins or indeed the same protein in its wild type form and a mutated protein structure. For clearly distinct protein structures, standard approaches for their comparisons such as the use of the RMSD (root mean squared deviation) may work reasonably well but providing a single parameter only for structure comparisons may not always be useful or sensitive enough to distinguish subtle structural changes involving, for example, single point mutations or a small number of amino acid substitutions. It should also be mentioned that while sequence comparison methods are rapid and reliable, since there is no general solution to the protein folding problem, sequence comparisons are insufficient by themselves to inform us about subtle structural changes that can distinguish between highly similar protein structures.

Some experimentation has already been undertaken to classify proteins according to their states. Tsuda et al. adopted a Support Vector Machine (SVM) classifier for fast protein classification [5]. They obtained 13 classes and reached an accuracy of about 90%. Weston et al. [6] used a semi-supervised classification with a kernel cluster and reached a result of 94.3%. Another interesting result has been obtained using a random forest approach and fifteen different supervised methods with about 11,000 pairs of protein domains leading to an accuracy of 97.0% [7]. Our focus in this paper is on accurate differentiation between structurally-similar proteins, which is a much harder problem to solve than comparing vastly different protein structures. Many cases of protein families can be found and it is important to be able to find characteristic features distinguishing proteins belonging to the same family. This could be valuable with respect to their functional roles in cell biology as well as potential applications in rational drug design.

One of the most important proteins abundantly expressed in all eukaryotic cells is the family of tubulin proteins, which is studied in this paper as a challenging test case for this methodology. It is also highly homologous with its bacterial ancestor, FtsZ, which will also be used here for comparison. We should stress again that comparing protein sequences is a trivial problem in bioinformatics while 3D structural features of folded proteins pose a much greater challenge, which is addressed here.

In the computational experiment reported below SVM was used because the quantity of data tested was relatively low, and a deep learning approach requires large data sets to achieve a high level of confidence. The novelty of our approach rests with the feature extraction using geometrical descriptors and its general applicability to 3D structure characterization, because geometric feature surfaces were used with significant results in many other applications before, e.g. [8, 9]. We believe that the classification provided here can be further improved with more data, more classes and a complex neural network. A complex neural network is one of the applications we are planning to implement in the near future. We intend to use a convolutional neural network to minimize the cost function to cluster the inputs correctly, because this could be an efficient way to find a pattern in the input data and it can be a significant improvement for our objectives. All of which is planned for future work, especially within the context of geometric deep learning [10], which nowadays is the state-of-the art of classification.

Tubulin is a key cytoskeletal protein, which has been exhaustively studied for its applications in several fields, including (i) being the target for various anti-cancer drugs [11] and (ii) the discrimination of the Saccharomyces complex [12]. It is a globular protein with a molecular weight of 55 kDa per monomer and its numerous isotypes expressed by separate genes have a broad distribution in animal and plant cells [13]. Tubulin is a building block of microtubules (MTs) and its stable form is an αβ -heterodimer. MTs play various important roles in all eukaryotic cells including cell motility, material transport and most importantly cell division where MTs form mitotic spindles [12-13].

The novelty of the present work rests with the application of geometrical descriptors coming from the field of face analysis to the classification of surfaces of proteins, with the aim of adopting this geometrical information as descriptive features and discriminating elements to classify proteins. Here, we test the method on the examples of tubulin isotypes and related proteins (e.g. FtsZ). The method can, of course, be applied to an arbitrary protein or indeed a protein complex but being able to discriminate between highly homologous proteins based on the geometrical shapes of their surfaces opens the door to numerous applications across the field of protein science. The idea comes from the realization that geometrical properties can well describe the surface of a 3D object such as a protein and could identify characteristic features when comparing two or more similar structures. Proteins surfaces can be split into two outer surfaces by cutting a plane through the data set including the main axis of rotational symmetry. These two halves of the outer surface, similarly to human faces, differ from one another depending on the protein type, and also can change their conformational states dynamically, similarly to human facial expressions. Thus, what in the field of pattern recognition is called face recognition could be transferred to the context of protein classification according to the typology. These common points have fostered the interest of uncovering the potentiality of cross-fertilization between these two fields with the aim of better categorization.

All eukaryotic organisms carry multiple genes coding for α and β tubulin (and other variants, e.g. γ), which are referred to as isoforms when comparing tubulin expressed by different organisms. When a single organism is discussed, various tubulin genes code for what are called tubulin isotypes. Isotypes have highly homologous amino acid sequences that appear to have diverged as a result of accumulated mutations since their separation by distinct speciation events [14]. Amino acid sequence similarity is very high for all tubulin proteins both within and between diverse species making structural comparisons difficult. At the cellular level, the roles of the α and β tubulin isotypes are essential, a result of subtle structural variations within their sequences [15] Several isotypes of the α and β tubulins have been identified in human cells, their existence and distribution providing a link to their specific roles in the polymerization and stability of MTs, among other roles [8] making structural differences correlate with functional roles in cells, importantly including cancer cells. For example, βII tubulin has been a common target for chemotherapy drug action and is involved in protein-protein interactions [2]. Hence again, the structural differences between tubulin isotypes significantly assist in drug design targeting specific isotypes such as βIII, which is overexpressed in all cancer cells. Through a search of available protein sequence databases, a total of ten unique β tubulin isotypes can be found, all of which have highly similar amino acid sequences and are generally well conserved. Sequence alignment, similarity and identity values of the studied isotype proteins (see below for details) range between 78% and 98%, indicating a major level of similarity between these structures. The question that remains is how do these sequence variations translate into structural differences.

As stated above, MTs are dynamic cytoskeleton polymers present in all eukaryotic cells made up of the protein tubulin. FtsZ is a close structural homologue of tubulin within prokaryotic cells, and plays an important functional role during bacterial cell division. A close relationship between FtsZ and tubulin can be seen from their very similar protein structures (Figure 1a). Both α and β tubulin share an approximate 35% sequence identity with FtsZ [16]. Both FtsZ and tubulin can assemble to form straight filaments. This association is regulated by guanosine triphosphate (GTP), which is bound in the junction between adjacent monomers (Figure 1b). FtsZ forms long protofilaments consisting of a single string of FtsZ proteins in contrast to tubulin, which makes cylindrical MTs. Unlike tubulin, FtsZ does not appear to provide a structural role throughout the bacterial cell cycle, but instead just plays a structural role during bacterial cell division, when it forms a band, known as the Z-ring, around the inner cell wall at the location where the cell will divide.

**Figure 1**

The main goal of the research reported here has been to investigate the following issues:

- whether it is possible to rely on features coming from the field of pattern recognition and face analysis to geometrically describe (and classify) the geometrical properties of the protein surface;
- whether it is possible to recognize different isotypes of the same protein from a different set of molecular dynamics snapshots;

- whether it is possible distinguish between two highly structurally similar but not identical proteins such as tubulin and FtsZ, and whether it is possible to distinguish arbitrary proteins with no relation to each other.

It is worth stating in this context that in general the main goal of a classifier is to separate objects belonging to different classes using a number of possible linear separators as shown in the examples presented in Figure 2.

**Figure 2**

It is reasonable to expect that using one of these separators one can get a datum that is on the other side of the hyperplane, which would then be misclassified because the hyperplane is really near  the ham data [17]. SVM is able to find a solution with a larger margin for the two-separator classifier as shown in Figure 2(a). This hyperplane works better than others as it is expected to reduce the number of misclassifications, because it is the one with the highest margins from the two sets of data.

The first part of this paper describes the development of the dataset using tubulin isotypes and FtsZ protein as test cases. Then, geometrical descriptors are computed on the 3D surface of these proteins. They are then converted into histograms and saved in a file. This file is the input of the classifiers. The code is provided in a pCloud repository [18]. The entire process is summarized in Figure 3.

**Figure 3**

This paper is organized as follows. In Section 2 geometrical descriptors used for implementing the feature extraction are described. Section 3 is the core of the paper and it outlines feature extraction and classification methods with a detailed description of the strategies and techniques performed. Section 4 summarizes and discusses the results comparing them with the-state-of-art results. Finally, Section 5 summarizes the work and discusses future developments.

## 2. Geometrical descriptors

The surfaces representing both human faces and proteins are geometrically considered as a free form. Thus, features coming from the field of differential geometry can be applied in order to understand their local and global properties. Geometrical descriptors are widely used in the area of 3D face recognition with significant results reported elsewhere in the literature [19, 20]. They underline different characteristics of a free-form and are an important tool for feature extraction [21] within the context of face analysis [22]. In this work, for the first time we apply these descriptors to proteins and use them for structural classification purposes [19].

The geometrical descriptors used in this research are the following geometrical descriptors [22, 23]: mean curvature ($H_{mean}$), principal curvatures (${k_1}_{mean}$ and ${k_2}_{median}$), the shape index ($S_{mean}$), the third coefficient of the second fundamental form ($g_{mean}$ and *sing*), and a descriptor enlightening the symmetry property ($F_{den2}$). Considering that these descriptors rely on the derivatives of the surface ($h_x$, $h_y$), they well describe the changes in surface curvature (${k_1}_{mean}$, *sing*, ${k_2}_{median}$, $g_{mean}$, $H_{mean}$), depressions and peaks (local minima and maxima) of the surface (${k_1}_{mean}$, *sing*, ${k_2}_{median}$, $g_{mean}$, $H_{mean}$), the shapes in terms of the types of surfaces ($S_{mean}$), and the surface's symmetry property ($F_{den2}$,*)*. These parameters are highly informative of the investigated surface's geometrical properties. Each descriptor can underline a specific characteristic of a certain surface. These descriptors are briefly described below in regard to their conceptual order. The first and second fundamental forms provide the first six descriptors of the set. They are used to measure distance on surfaces and are defined by the formula

$$ds^2 = Edu^2 + 2Fdudv + Gdv^2 \ (1)$$

where *E, F, G, e, f* and g are their coefficients given by:

$$E = 1 + h_x^2, \ (2)$$

$$F = h_x h_y, \ (3)$$

$$G = 1 + h_y^2 \ , \ (4)$$

$$e = \frac{h_{xx}}{\sqrt{1+h_x^2+h_y^2}}, \ (5)$$

$$f = \frac{h_{xy}}{\sqrt{1+h_x^2+h_y^2}}, \ (6)$$

$$g = \frac{h_{yy}}{\sqrt{1+h_x^2+h_y^2}}. \ (7)$$

Curvatures are used to measure how a regular surface *x* bends in. If D is the differential and N is the normal plane to a surface, then the determinant of DN is the product of the principal curvatures, and the trace of DN is the negative of the sum of principal curvatures. At point P, the determinant is the Gaussian curvature K of x at P. The negative of half of the trace of DN is called the mean curvature H of x at P.

The principal curvatures $k_1$, $k_2$ are the roots of the quadratic equation given below:

$$x^2 - 2Hx + K = 0 \ (8)$$

Thus, we can choose $k_1$ and $k_2$ so that:

$$k_1 = H + \sqrt{H^2 - K} \text{ and } k_2 = H - \sqrt{H^2 - K} \ (9)$$

where

$$K = \frac{eg-f^2}{EG- F^2} \text{ (10)}$$

$$H = \frac{eG - 2fF + gE}{2\,(EG - F^2)} \text{ (11)}$$

In terms of the principal curvatures, Gaussian (*K*) and mean curvatures (*H*) can be written as

$$K = k_1 k_2, \text{(12)}$$

$$H = \frac{k_1+k_2}{2} \text{ (13)}$$

where *h* is a differentiable function representing the three-dimensional surface.

The shape index *S*, which describes the shape of the surface, is defined as [24]:

$$S = -\frac{2}{\pi} arctan \frac{k_1+k_2}{k_1-k_2}, S \in [-1,1], k_1 \leq k_2. \text{ ( 14)}$$

Some descriptors highlight particular facial lines, such as $F_{den2}$, which shows visible facial part contours. It can be computed using the formula:

$$\frac{\boldsymbol{F}}{\boldsymbol{1 + (h_x)^2 + (h_y)^2}}, \boldsymbol{(15)}$$

where:

- *h* is the differentiable function z = h (x, y) representing the face/protein surface;
- $\boldsymbol{h_x}$ and $\boldsymbol{h_y}$ are the first derivatives of *h* with respect to *x* and *y* [25].

In a protein, Fden2 can underline different trends of the free form analyzed. In particular, this descriptor has high and low values in correspondence to concavities and convexities, and values approximately equal to zero on critical points.

The surfaces of human faces are given by depth maps, which are manageable as matrixes (X Y Z). For each coordinate pair X, Y, there is a unique value of Z. Since proteins do not have a default form, their surfaces are split up in two parts divided into two opposite faces: surfaces with a positive Z-axis and those with a negative Z-axis in order to yield two shells that complete the protein surface.

The descriptors used are mapped onto the surfaces as described in Section 3.4. These descriptors are calculated for all protein faces considered in the following. An example of $F_{den2}$ applied to both a human face and a protein is shown in Figure 4a. The descriptor *sing* is built from the application of the sine standard function applied to the third coefficient of the second fundamental form ($g$) (see Figure 4b) [23]. Mean and median filters have been applied to the primary descriptors $S$, $k_1$, $k_2$, $g$, and $H$. Mean and median values are computed in squared neighborhoods of side 5 around each point of the facial depth maps [23]. These descriptors are labelled as follows: $S_{mean}$, (see Figure 4c), $k_{1_{mean}}$ (see Figure 4d), $k_{2_{median}}$ (see Figure 4e), $g_{mean}$ (see Figure 4f) and $H_{mean}$ (see Figure 4g).

**Figure 4**

## 3. Material and methods

At the beginning of this section we give a brief introduction to some basic concepts related to Machine Learning, which can be useful for understanding the methods used in this paper. Machine Learning (ML) is a subset of Artificial Intelligence (AI) tools that include mathematical and statistical models, which complete tasks with experience gained through training. The quality and amount of the training data have an important role in this process. ML classifiers can be divided into two types based on their training methods:

supervised and unsupervised learning. Supervised learning needs a training phase with labeled training data (i.e. sample data containing input-output pairs) in order to learn the relationship between the input and output data. On the other hand, unsupervised learning algorithms do not employ labeled training data and they aim to divide the dataset into clusters without the training phase. In this work, we use a discriminative model (a supervised model) that employs Support Vector Machine (SVM). The aim of this model is to determine the division of different clusters without considering how data are generated, unlike generative models, which do consider how the data are generated during the process. In our model, dot-product kernels are used to compute the similarity between two vectors in a higher dimensional feature in a more efficient manner. For the SVM, we tried both linear and non-linear kernels. As the linear kernel essentially performs the normal dot-product, the similarity score is calculated as the length of the projection of one vector onto another. The non-linear kernel can perform the dot product in a higher dimensional feature space. Even though non-linear kernels may be slower to use due to the computational complexity, they usually yield more favorable results. Geometric Deep Learning is a new field in deep learning that aims to build neural networks that can learn from non-Euclidean data, for example from graphs or complex surfaces.

The process we follow in this paper starts with the collection of protein data. In the present example we focus on tubulin whose bovine structure has been crystallized and can be found in the Protein Data Bank (PDB). However, its various isotypes have not been crystallized and hence these structures need to be generated by homology modeling using the bovine (not human) variant of this protein as a template. To obtain frames of the protein structure, it is necessary to run MD simulations for some time, typically 10-100 nanoseconds and take snapshots, approximately every nanosecond, at the very moment when the structure relaxes to an equilibrium conformation. Only the atoms comprising the protein are kept in the file used for these MD simulations with the ligand atoms removed in order to avoid false representations of the protein since ligands are not part of the protein and can form an occlusion during the process of protein recognition. The next step in this computational experiment is to analyze similar but not identical proteins and their states, for example tubulin isotypes with each other or a tubulin isotype and FtsZ and to compare the two for similarities and differences.

The result of these MD simulations is in each case a PDB-formatted file that is a 3D representation of a protein, which is converted into a MAT file using a MATLAB script. In the current work several software packages are used: Matlab 9.5 (R2018b) [26] for the feature extraction using geometrical descriptors, Anaconda 1.9.6 [27] with Python 3.7 [28] and the library sklearn 0.22 [29] for the implementation of classification methods and R-3.5.3 for the k-means algorithm [30].

### 3.1. Molecular dynamics simulations

The tubulin crystal structures available in the PDB are those for bovine protein. The bovine tubulin structure of tubulin (PDB ID: 1JFF) [31] was used as a template to construct the homology model for human αβ tubulin isotypes (βI (UniProtKb: P07437), βIIa (UniProtKb: UniProtKb: Q13885), βIIb (UniProtKb: Q9BVA1), βIII (UniProtKb: Q13509), βIVa (UniProtKb: P04350), αβIVb (UniProtKb: P68371), αβV (UniProtKb: Q9BUF5), αβVI (UniProtKb: Q9H4B7) and βVIII(UniProtKb: Q3ZCM7)) using the Molecular Operating Environment (MOE) software package [32]. Multiple sequence alignment results contained in Figure 5 show that human β-tubulin isotypes exhibit residue composition variations at different locations.

**Figure 5**

Sequence similarity matrix and sequence identity matrix of the tubulin isotypes are shown in Figure 6(a) and (b), respectively. The matrix values (i, j) for the percentage identity and similarity metrics are equal to the number of sequence matches between chains i and j, divided by the number of residues in chain i. Residues are considered identical if their single-letter code is the same (note that MSE-Selenomethionine and MET-Methionine are considered "identical"). Residues are "similar" if their BLOSUM62 substitution score is greater than zero.

**Figure 6**

The atomic coordinates of similar but not identical FtsZ dimer were obtained from the Protein Data Bank as (PDB ID: 1W5B) [33]. The coordinates for the missing residues of the proteins were obtained by modeling using the MOE package [32]. Since the C-terminus has not been included in the electron crystallography data for the tubulin structure, we did not consider it in our calculations. The missing hydrogens for heavy atoms were added using the tLEAP module of AMBER [34] with the AMBER14SB force field. The protonation states of all ionizable residues were determined at pH = 7 using the MOE program. Each protein model was solvated in a 12 Å box of TIP3P water. Na+ and Cl− ions were added in order to bring the salt concentration to the physiological value of 0.15 M. After minimization, the MD simulations were carried out in three steps: heating, density equilibration, and production. First, each solvated system was heated to 300 K for 50 ps, with weak restraints on all backbone atoms. Next, density equilibration was carried out for

50 ps of constant pressure equilibration at 300 K, with weak restraints. Finally, MD production runs were performed on all systems for 100 ns. Ligands and ions were all removed from the complex after equilibration in order to avoid false representations of the protein since ligands can form an occlusion during the process of protein recognition. After equilibration, density-based clustering algorithm from the AMBER software was used for cluster analysis of MD trajectories (20). Several snapshots from top clusters were selected for all further calculations in the study.

The result of our simulation is a PDB-formatted file (a 3D representation of all atoms comprising the protein), which is converted into a MAT file using a MATLAB script.

### 3.2. Data augmentation

To expand the dataset for FtsZ, a data augmentation technique is used where each structure is rotated around the Z-axis in 40° steps. Subsequently, the 3D protein representation is ready to be used for feature extraction. It was not necessary to follow the same procedure for tubulin since we have many examples available. The purpose of reorienting the z-axis is not only to obtain additional examples, but also in order to not have a bias inside the classifier, in fact most of the rotated proteins were used during the test phase. Both hemispheres of the protein were used to have a complete dataset. Then, to avoid the over-fitting problem a k-fold cross validation is implemented with k = 5.

Cross validation is a powerful technique used to avoid overfitting. When the model is trained and tested on the same dataset, high scores can be easily obtained since the model becomes biased. In this case, low score results are obtained when the model is tested on an unseen dataset. Using cross validation, the dataset is divided into k sub parts, called folds. Then, the training is performed iteratively on the k-1 folds and the remaining fold is used for the testing phase. In this way, the test set will be a truly unseen dataset for the model. One such example is shown in Figure 7 (https://probis.nih.gov/) [35].

**Figure 7**

At this point the 3D protein representation is ready and the feature extraction can be performed.

### 3.3. Protein samples

In this computational experiment, we used a total of 889 examples of tubulin structure files for 9 isotypes, as shown in Table 1.

**Table 1**

Using data augmentation, the 13 FtsZ protein samples were rotated in order to create 65 samples, most of them used only during the test phase. The binary classification between tubulin and FtsZ was performed using the samples shown in Table 2.

**Table 2**

### 3.4. Data processing

The x-, y- and z-coordinates were extracted from the PDB file. First, the data were shifted in order to be geometrically symmetric with respect to x-, y- and z- axes, i.e. the center of the coordinate systems is the geometric center of the dataset: $(x, y, z) \rightarrow (x - \triangle x, y - \triangle y, z - \triangle z)$ where $\triangle x = (xmax - xmin)/2$, $\triangle y = (ymax - ymin)/2$ and $\triangle z = (zmax - zmin)/2$. Then, the data were divided into two groups of positive and negative z-values. Finally, for each group, the exterior surface with a desired resolution was calculated using "meshgrid" and “griddata” commands in Matlab with the cubic interpolation method.

The descriptors were mapped onto the surfaces as follows. The surfaces were given by point clouds where points are non-connected (not a mesh) and arranged in a square grid.

This type of data is called depth map and can be described by matrices: X, Y, Z, where Z is the one describing the "surface" and is represented in these formulas as $h$. Through Matlab "gradient" function, the derivatives *hx, hy…* were evaluated so that other matrices representing the first derivative with respect to *x*, the first derivative with respect to *y*, etc., were generated and stored. Then, the implementation formulas for the descriptors were calculated on the matrices previously computed and new matrices were obtained representing every geometrical descriptor.

For each protein the Z axis was divided in two files: one for the positive part and the second for the negative part using the formula: z – max(z) + (|max(z)-min(z)|)/2 . Each part represents a "face" of the protein and the geometrical features were computed for both the faces. Then, for every geometrical descriptor a 9-bin histogram was created with the same equidistance for the X-axis.

The MATLAB code loaded all data and the following processing steps were performed for all the datasets:

- the class of the protein was extracted from the filename and the class was recorded in the first column of the dataset matrix;
- geometrical descriptors were computed from matrix Z (positive and negative);
- histograms were created and each bin was written in the right column of the dataset matrix;
- at the end of each loop the dataset matrix became the input for the classifier.

The entire process is summarized in Figure 8.

**Figure 8**

In this computational experiment, 9 isotypes were used (indeed, the classifier will work with 9 classes). The classes were chosen 1 to 9 in an ascendant order as shown in Table 3.

**Table 3**

This task was performed using a switch case construct. The right class was written in the first column of the Features Matrix.

### 3.5. Feature extraction

For every geometrical descriptor, a 9-bin histogram was created. Since it is possible that some descriptors have values $\in \mathbb{C}$ (complex), a check was performed first. The geometrical descriptors were calculated using 9 bins and the X-axis values were compressed between -0.2 and 0.2, then the Y-axis values were saved and used as features. Some examples of histograms are shown in Figure 9.

**Figure 9**

Finally, when all descriptors for all protein data were computed, the resultant matrix was copied into a file. For tubulin and other proteins these descriptors can underline specific characteristic of a certain surface. They can indicate different trends of the free form analyzed and they can describe the shape of the surface. The features are extracted with multiple geometrical descriptors to extract more details; using this approach, also small differences in convexity and concavity can be recognized during the classification. Analyzing the features extracted, the most important features were found from parameter values of *Fden2* and *sing*, because analyzing the data these values were sufficiently different to help the classifier select the right class. In particular, *Fden2* is meant to be descriptive for the its behavior in the loci of critical points, and *sing* for curvature changes, local minimums in convexities and local maximums in concavities, respectively.

### 3.6. Classification

The adopted classifiers were k-means and SVM. First, an unsupervised method was tested (k-means) using 9 clusters and a limited number of iterations, then a supervised method (SVM) using linear and non-linear kernels was used. In these cases, it is not a simply binary classification, but there are many classes (9) and many features (more than 100), so some distributions cannot separate the dataset in a linear way or with a linear separator as a high misclassification rate is reached. An interesting improvement is to use a non-linear separator or a kernel trick. An example of a non-linear kernel is the RBF kernel, which in this test led to positive results.

A linear and a nonlinear kernel (RBF in our case) were chosen in order to see whether a non-linear kernel can reach better results. The difference between linear and non-linear kernel is on the way they divided dataset into classes. A linear kernel uses a linear function to divide it and it is less time consuming but also less precise. A non-linear kernel uses a non-linear function, so it can divide the dataset better. The cross validation has not been performed here because the results were positive, and hence we have already avoided the overfitting problem. The validation part was performed using a large number of parameters and the best ones were selected for the testing part.

#### 3.6.1. k-means

An unsupervised approach was performed using a k-means classifier implemented in R. The matrix file was loaded and the column with the label was deleted. Then, the classifier was tasked with finding 9 clusters in the input data and at the end there was a comparison made between the clustering and the right label.

k-means works in an iterative way and it performs three steps. In the first step, the dataset is loaded, and the number of clusters is chosen. The centroids are created in a random position. In the second step, each data point is assigned to a nearest cluster. The range for the initialization of the centroids of k-means is set from 2 to 10. The Euclidean distance is computed between a point and every centroid. The minimum distance centroid is chosen as the following cluster:

$$argmin\ dist(c_i, x)^2,$$

where c is the centroid and x the data points. In this last phase the centroids are computed again as the mean of all the data points of the cluster:

$$c_i = \frac{1}{|S_i|} \sum x_i,$$

where S_i is the sum of a single cluster. Therefore, new centroid positions are computed, and this loop continues until the centroid positions do not change significantly.

The stop condition is given by the following criteria:

- no data points change the cluster;
- the sum of distances is at the minimum;
- the maximum number of iterations is reached.

Therefore, when the convergence is obtained the algorithms stops.

The final result achieved in this example was 76.6%, which is an acceptable result, considering that it is an unsupervised method. Nonetheless, in order to improve the method’s accuracy, other types of classifications were tested by us and we discuss them below.

### 3.6.2. Support Vector Machine

The first test was performed using a linear kernel where λ is a key parameter of SVM. In fact, the main factors in SVM are setting a large margin and reducing the misclassification rate. These two properties are inversely proportional, and the λ parameter helps to find a trade-off. A large value of λ is for a small margin, whereas a small value of λ is for a large margin. The right λ parameter depends on the test data. The steps used are as follows:

- the dataset is loaded and features and labels are divided;
- the dataset is randomly split into 60% training set, 10 % validation set and 30% test set;
- the training is performed using a linear kernel. We then use different values of λ in the range $10^{-5}$ *to* $10^{5}$and it is evaluated on the validation set. The best parameter found on the validation set is λ = $10^{-5}$ with a score of 95.1%;

the model is tested and scored on the validation set with the best parameters.

The accuracy obtained changes using different λ values. As a matter of fact, by increasing the λ value, the optimization will choose a smaller margin hyperplane, but the best parameters depend on the dataset and in this case the best value is obtained as λ = $10^{-5}$. The final evaluation on the test set with the best parameter λ = $10^{-5}$ was found to be 92.4%.

The dataset was built using 9 different Tubulin isotypes. Hence, the number of classes used for the SVM classifier was 9; the same number was used in the k-means test, in order to have comparable results. The confusion matrix is an important tool to evaluate the results, since it gives precise information about misclassification. A confusion matrix without normalization and a normalized confusion matrix are represented in Figure 10. It this case, the accuracy is very high, since there is misclassification found only in one class.

**Figure 10**

The second test was performed using an RBF kernel. The number of features used was 112 and the dataset was not large, so an approximation of the RBF kernel was not taken into consideration (22). The steps used are as follows:

- the dataset is loaded and features and labels are divided;
- the dataset is randomly split into 60% training set, 10 % validation set and 30% test set;
- the training is done using an RBF kernel. We then use different λ and gamma parameters in the range between $10^{-5}$ to $10^{15}$ and it is evaluated on the validation set. The best parameters on the validation set are found to be: λ = 100 and gamma = $10^{-9}$ with a score of 98.0%;
- the model is tested and scored on the validation set with the best parameters.

Note that the achieved accuracy changes significantly using different λ and gamma values. The gamma parameter that is used in the RBF kernel function is the inverse of the standard deviation of the RBF kernel, which is used as a similarity function. A small value of gamma indicates a large variance where two points can be matched as similar. This results in a smoother decision-making by the model. A higher gamma value has the opposite effect on the process. The challenge will be to find an optimum value of gamma for the given data set. Indeed, by increasing the λ value, the optimization will choose a smaller margin hyperplane, but the best parameter depends on the dataset selected and, in this case, the best is 100. The final evaluation on the test set with the best parameter λ = 100, gamma = $10^{-9}$ and the accuracy obtained was 96.5%.

The same methodology was applied to tubulin and FtsZ classifications.

## 4. Results and discussion

In the case of tubulin isotype comparison, the best result was given by the SVM classifier with an RBF kernel. All results are summarized in Table 4.

**Table 4**

In the case of tubulin and FtsZ comparison, the best result is also given by the SVM classifier with an RBF kernel. All results are summarized in Table 5.

**Table 5**

These results are competitive with the state-of-the-art results found in the literature. A fast protein classification method [5] based on an SVM classifier reached an accuracy of about 90% with 13 classes. Another study [7] used a semi-supervised classification with a kernel cluster and achieved a 94.3 % accuracy. Consequently, the results of the present study appear to be significant. This work is a starting point toward protein classification based on geometrical features and we expect that even better results can be reached in the future. A natural continuation of this work can be to study important features of a protein, for example characterization of a binding pocket [36] for a ligand, a catalytic domain recognition or a protein-protein interaction interface.

A larger experiment was performed using several additional proteins in order to provide an increased validation for the method proposed in this paper. This test involved four arbitrarily chosen FtsZ protein structures, namely: 2R6R, 2VAW, 2VAP and 2VAM. These structures correspond, respectively, to the following biological species: *B. subtilis,* *Pseudomonas aeruginosa, M. jannaschii and Aquifex aeolicus*. In this test 683 samples were used as listed in Table 6.

**Table 6**

The results of this test are very encouraging as shown in Table 7, which summarizes the use of various classifiers for different tests performed and their accuracy levels achieved.

**Table 7**

To avoid over-fitting and to generalize the method in a better way a 5-fold cross validation is performed. In this way the classifier is not biased by the test set and it also works well with other proteins. The last experiment showed that it also works well with four very different proteins. In this test a k-cross validation method was applied using k=5.

## 5. Conclusions

A novel method for protein characterization and classification has been proposed in this paper, which is inspired by and uses the algorithms from the facial recognition field. The first application of this method involves a challenging case of classification of highly homologous tubulin isotypes using as features some geometrical descriptors typically found within the context of face recognition analysis. While human faces and proteins represent very different biological structures, they are both free-form surfaces and the same types of geometrical features are adopted for their classification and recognition.

The aim of this study has been to implement different classifiers to be tested on the dataset previously built. In this work we used the following approaches: SVM with a linear RBF kernel, and a k-means algorithm. This methodology and the geometrical descriptors have been used for protein classification. The first classification was performed using the tubulin protein and 9 of its isotypes. The second application performed used two structurally similar proteins: bovine tubulin and FtsZ and third application involved four unrelated proteins. In all cases very encouraging results were obtained.

It should be stressed that until now the use of RMSD as a measure of similarity has been prevalent in protein biophysics, especially regarding structural comparisons. However, this approach relies on a single number, which does not allow for feature extraction or more detailed shape comparisons, which the present methodology provides. A single parameter such as an RMSD value can answer the question if two proteins are structurally similar or not but does not address the issue regarding which features differ between them. For this reason, our method can assist in identifying structure-function dependence when comparing various proteins, even highly similar ones. Since we only investigate geometrical features, both physical and chemical properties are not directly involved in our method but can eventually be extracted by mapping geometrical features back onto to amino acid distributions underlying them. Also, the number of potential mutations of any protein, in particular tubulin, is astronomical. Consequently, brute force methods are not viable in classifying the role of specific mutations regarding the root causes of the conformational changes resulting in dysfunction of a given protein. However, our methodology based on machine learning approaches may offer a viable alternative with numerous potential applications in protein biophysics and beyond.

In this study, MD has been used to generate additional models of each protein for the training purpose where each of the models is extracted from equilibrated MD trajectories after clustering. Clustering of the trajectory provides us with different conformations of the same protein from MD trajectories. We used several snapshots from each structural cluster, which makes it possible to probe diverse sampling of the trajectory.

In future work, a larger set of protein structures will be used to address the issue of structural diversity across the entire PDB dataset consisting of over 150,000 entries.

The results obtained and reported here are significant: a 96.5 % accuracy for tubulin isotype classification, a 98.2 % accuracy for tubulin and FtsZ classification and a 98% accuracy for a set of four arbitrarily chosen protein structures. SVM is a classifier with competitive performance using a small dataset (< 3000 samples) and in this case the results are significant. The application of a neural network can be a future development using a convolutional type on a larger dataset (> 10,000 samples). The conclusion is that these geometrical descriptors work properly with the description of protein surfaces and they are accurate enough to properly describe protein surfaces.

Several future developments can be taken in consideration, namely:

- building a database adding more samples and more proteins;
- computing more features and testing classifiers, using more geometrical descriptors and filters;
- applying our method to different data set for the purpose of protein classification such as Hemoglobin classification [reference: Clang et al. ]
- developing more data augmentation techniques to enlarge the dataset;
- identifying specific important features on a protein, for example a binding pocket for a ligand or a protein-protein interaction interface.

Other important improvements will be performed in future tests. First, we will employ neural networks that were applied here with significant results with 3D geometrical descriptors [19]. Second, using a large dataset with unnecessarily numerous features the classifier could be slow, so some feature optimization techniques will be implemented in order to [37] accelerate the training of the kernel machine.

**Author Contributions:** For research articles with several authors, a short paragraph specifying their individual contributions must be provided. The following statements should be used “conceptualization, F.M. and JAT; methodology, FM, JAT and LDG; software, LDG; validation, LDG; formal analysis, F.M. and JAT ; investigation, FM, JAT and LDG ; Matlab scripts, VR; resources, MA and EV; data curation, MA; writing—original draft preparation, LDG and MA; writing—review and editing, MA, F.M. and JAT; visualization, LDG and MA;  supervision, F.M. and JAT; project administration, JAT; funding acquisition, JAT and EV”, please turn to the CRediT taxonomy for the term explanation.

**Funding:** This research received no external funding.

**Acknowledgments:** Computational infrastructure of the Pharma-matrix cluster at the Cross Cancer Institute is gratefully acknowledged.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References:**

1. Gainza, P., et al., *Deciphering interaction fingerprints from protein molecular surfaces.* bioRxiv, 2019: p. 606202.

2. Planas-Iglesias, J., et al., *Understanding Protein–Protein Interactions Using Local Structural Features.* Journal of Molecular Biology, 2013. **425**(7): p. 1210-1224.

3. Rupp, B. and J. Wang, *Predictive models for protein crystallization.* Methods, 2004. **34**(3): p. 390-407.

4. Saberi Fathi, S.M., D.T. White, and J.A. Tuszynski, *Geometrical comparison of two protein structures using Wigner-D functions: Geometrical Comparison of Protein Structures.* Proteins: Structure, Function, and Bioinformatics, 2014. **82**(10): p. 2756-2769.

5. Tsuda, K., H. Shin, and B. Schölkopf, *Fast protein classification with multiple networks.* Bioinformatics, 2005. **21**(suppl_2): p. ii59-ii65.

6. Weston, J., et al., *Semi-supervised protein classification using cluster kernels.* Bioinformatics, 2005. **21**(15): p. 3241-3247.

7. Jain, P., J.M. Garibaldi, and J.D. Hirst, *Supervised machine learning algorithms for protein structure classification.* Computational Biology and Chemistry, 2009. **33**(3): p. 216-223.

8. Masci, J., et al. *Geodesic Convolutional Neural Networks on Riemannian Manifolds*. 2015. IEEE Computer Society.

9. Monti, F., et al. *Geometric Deep Learning on Graphs and Manifolds Using Mixture Model CNNs*. in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017.

10. Bronstein, M.M., et al., *Geometric Deep Learning: Going beyond Euclidean data.* IEEE Signal Processing Magazine, 2017. **34**(4): p. 18-42.

11. Espinosa, E., et al., *Classification of anticancer drugs—a new system based on therapeutic targets.* Cancer Treatment Reviews, 2003. **29**(6): p. 515-523.

12. Huang, C.-H., F.-L. Lee, and C.-J. Tai, *The β-tubulin gene as a molecular phylogenetic marker for classification and discrimination of the Saccharomyces sensu stricto complex.* Antonie van Leeuwenhoek, 2009. **95**(2): p. 135-142.

13. Ludueña, R.F., *Are tubulin isotypes functionally significant.* Molecular Biology of the Cell, 1993. **4**(5): p. 445-457.

14. Fitch, W.M., *Homology: a personal view on some of the problems.* Trends in Genetics, 2000. **16**(5): p. 227-231.

15. Richards, K.L., et al., *Structure–Function Relationships in Yeast Tubulins.* Molecular Biology of the Cell, 2000. **11**(5): p. 1887-1903.

16. Schlieper, D., et al., *Structure of bacterial tubulin BtubA/B: Evidence for horizontal gene transfer.* Proceedings of the National Academy of Sciences of the United States of America, 2005. **102**(26): p. 9170-9175.

17. Gunn, S.R., *Support Vector Machines for Classification and Regression.* p. 52.

18. https://u.pcloud.link/publink/show?code=XZwyRNkZdgxbscKvDcz9RcNn832cPYuD3pRV)

19. Ciravegna, G., et al., *Assessing Discriminating Capability of Geometrical Descriptors for 3D Face Recognition by Using the GH-EXIN Neural Network*, in *Neural Approaches to Dynamics of Signal Exchanges*, A. Esposito, et al., Editors. 2020, Springer: Singapore. p. 223-233.

20. Cirrincione, G., et al., *Intelligent Quality Assessment of Geometrical Features for 3D Face Recognition*, in *Neural Advances in Processing Nonlinear Dynamic Signals*, A. Esposito, et al., Editors. 2019, Springer International Publishing: Cham. p. 153-164.

21. Li, S.Z. and A.K. Jain, *Handbook of Face Recognition*. 2 ed. 2011, London: Springer-Verlag.

22. Marcolin, F., et al., *Three-dimensional face analysis via new geometrical descriptors*, in *Advances on Mechanics, Design Engineering and Manufacturing : Proceedings of the International Joint Conference on Mechanics, Design Engineering & Advanced Manufacturing (JCM 2016), 14-16 September, 2016, Catania, Italy*, B. Eynard, et al., Editors. 2017, Springer International Publishing: Cham. p. 747-756.

23. Marcolin, F. and E. Vezzetti, *Novel descriptors for geometrical 3D face analysis.* Multimedia Tools and Applications, 2017. **76**(12): p. 13805-13834.

24. Koenderink, J. J. and Van Doorn, A. J., *Surface shape and curvature scales. Image and vision computing*, 1992, 10(8), 557-564.

25. Vezzetti, E. and F. Marcolin, *Geometrical descriptors for human face morphological analysis and recognition.* Robotics and Autonomous Systems, 2012. **60**(6): p. 928-939.

26. *MATLAB*. 2018, The MathWorks Inc.: Natick, Massachusetts.

27. *Anaconda Software Distribution. Computer software.* 2019, Anaconda,.

28. Van Rossum, G. and F.L. Drake Jr, *Python*. 2019, Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands.

29. *Version 0.22.0 — scikit-learn 0.22 documentation* - https://scikit-learn.org/stable/whats_new/v0.22.html.

30. *Download R-3.5.3 for Windows. The R-project for statistical computing* - https://cran.r-project.org/bin/windows/base/old/3.5.3/.

31. Löwe, J., et al., *Refined structure of αβ-tubulin at 3.5 Å resolution11Edited by I. A. Wilson.* Journal of Molecular Biology, 2001. **313**(5): p. 1045-1057.

32. *Molecular Operating Environment (MOE). Group, Chemical Computing*. 2012: Montreal, QC, Canada.

33. Oliva, M.A., S.C. Cordell, and J. Löwe, *Structural insights into FtsZ protofilament formation.* Nature Structural & Molecular Biology, 2004. **11**(12): p. 1243-1250.

34. D.A. Case, et al., *AMBER 2014*. 2014: University of California, San Francisco.

35. Konc, J., et al., *ProBiS-CHARMMing: Web Interface for Prediction and Optimization of Ligands in Protein Binding Sites.* Journal of Chemical Information and Modeling, 2015. **55**(11): p. 2308-2314.

36. Saberi Fathi, S.M. and J.A. Tuszynski, *A simple method for finding a protein's ligand-binding pockets.* BMC Structural Biology, 2014. **14**: p. 18.

37. Rahimi, A. and B. Recht, *Random Features for Large-Scale Kernel Machines.* p. 10.

**Figures' captions:**

**Figure 1:** Structural similarities between tubulin and FtsZ proteins. The tubulin dimer consists of an $\alpha$-tubulin and a closely related $\beta$-tubulin monomer. $\alpha\beta$-tubulin heterodimers associate head to tail to form protofilaments and laterally to form the cylindrical MT wall. GTP and GDP nucleotides (ball and stick models) are bound to $\alpha$ and $\beta$ tubulin, respectively. (b) The FtsZ dimer consists of two identical monomers with GTP bound to N-terminals (blue). In both (a) and (b) N-terminals (blue) and C-terminals (red) are separated by H7 helices (green). N-terminal regions show the typical nucleotide-binding motif with parallel $\beta$ sheets connected by $\alpha$ helices known as the Rossmann fold. By comparing the two protein structures, the differences in C-terminal regions are obvious. GDP and GTP are shown in ball and stick models. The figures were rendered using the

MOE (Molecular Operating Environment) software. PDB ID for tubulin: 1JFF. PDB ID for FtsZ: 1W5B.

**Figure 2:** Valid solutions can be found with perceptron in a binary case(a) and the best theoretical solution that a SVM classifier can find (b).

**Figure 3:** Flow chart of the entire protein characterization and classification process.

**Figure 4:** Effects of applying different descriptors (a) F_den2 ,(b) sing (c) , $S_{mean}$ , (d) $k_{1_{mean}}$ ,(e) $k_{2_{median}}$ , (f), $g_{mean}$ ,and (g) $H_{mean}$to a human face (left column) and to the tubulin protein (right column)

**Figure 5:** Sequence alignment of β tubulin isotypes. Each of the human β tubulin isotypes that were identified in our screen of the UniProt databases were aligned using the MOE package. Prior to performing the alignment, the highly variable carboxy-terminal residues were removed from each sequence. This was done as the template structure, 1JFF, does not contain any of these residues. At each position within the alignment, dark blue boxes indicate identical residues; light blue boxes indicate residues that are conserved, while red boxes indicate residues that are divergent (poorly aligned).

**Figure 6:** a) Sequence similarity matrix and (b) sequence identity matrix of the studied tubulin isotypes. The matrices are heatmap color-coded (the darker the shade, the more similar the values are).

**Figure 7:** Tubulin protein image for two different rotations with respect to the Z-axis. The blue color-code represents not conserved and red color represents the more conserved as it shown in the scale bar. The images were taken from https://probis.nih.gov/.

**Figure 8:** Protein data processing overview. The input consists of a 3D structure of a protein from either the PDB database or from homology modeling combined with MD simulations. The color selection in the input structure is arbitrarily chosen for better visualization. The output consists of geometrical descriptor values obtained from a facial recognition algorithm.

**Figure 9:** 9 bin histograms calculated using (a) $F_{den2}$, (b) $g_{mean}$ and (c) $H_{mean}$ geometrical descriptor

**Figure 10:** Confusion matrix of SVM classifier using the RBF kernel.

**Figures:**

**Figure 1**

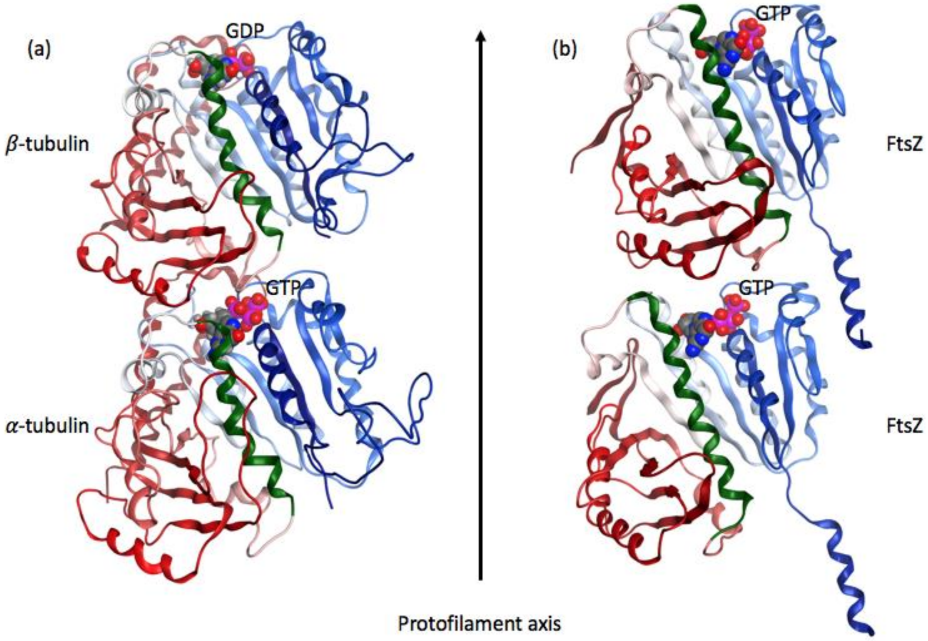


**Figure 2**

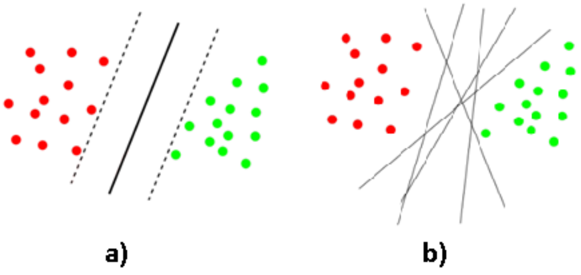
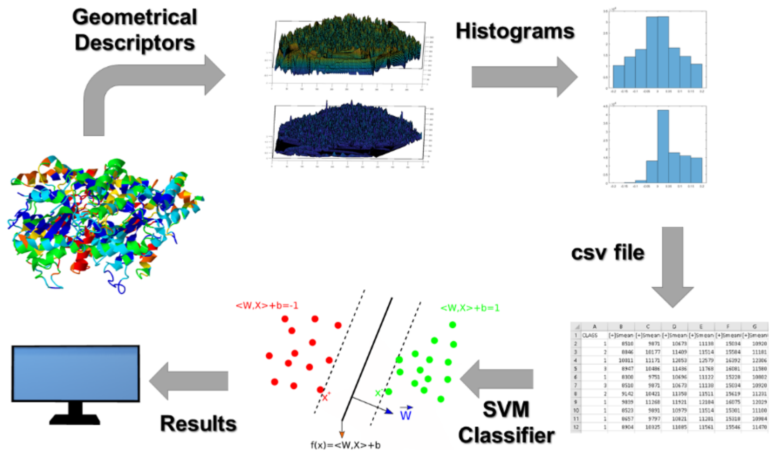
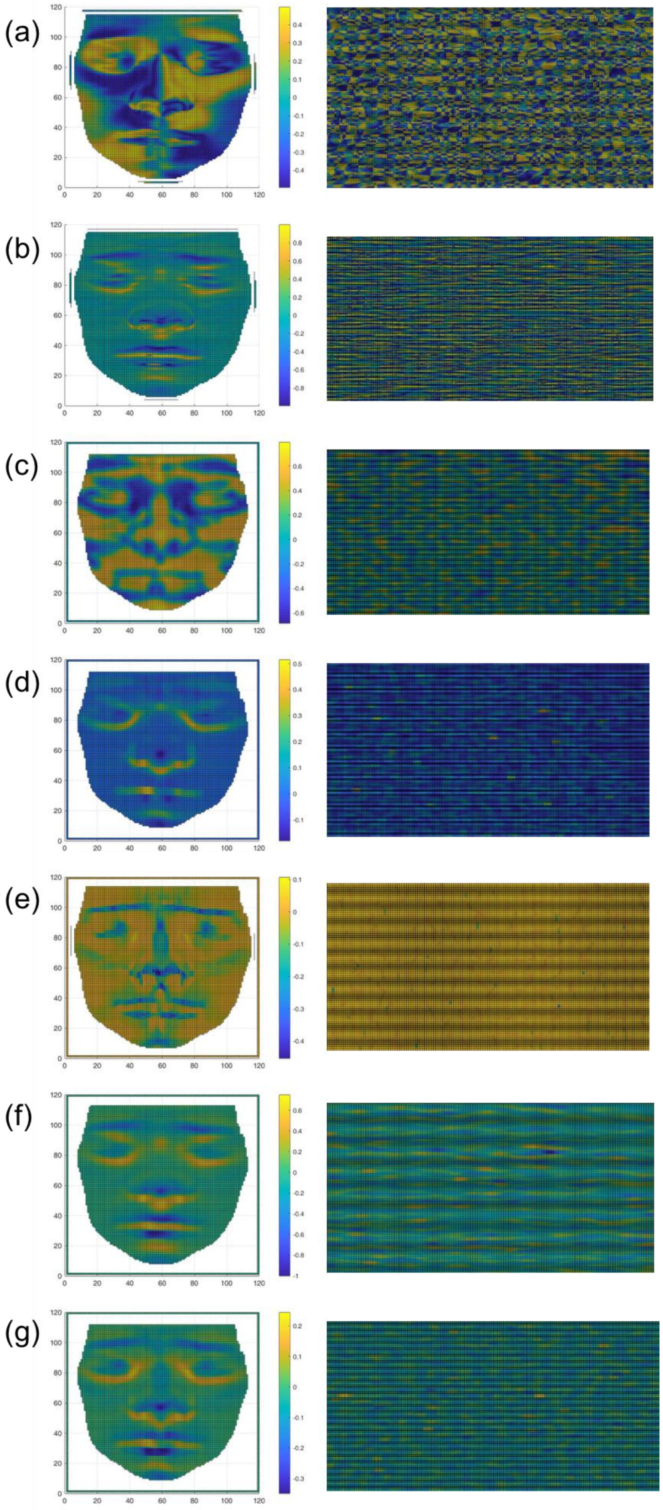


**Figure 3**

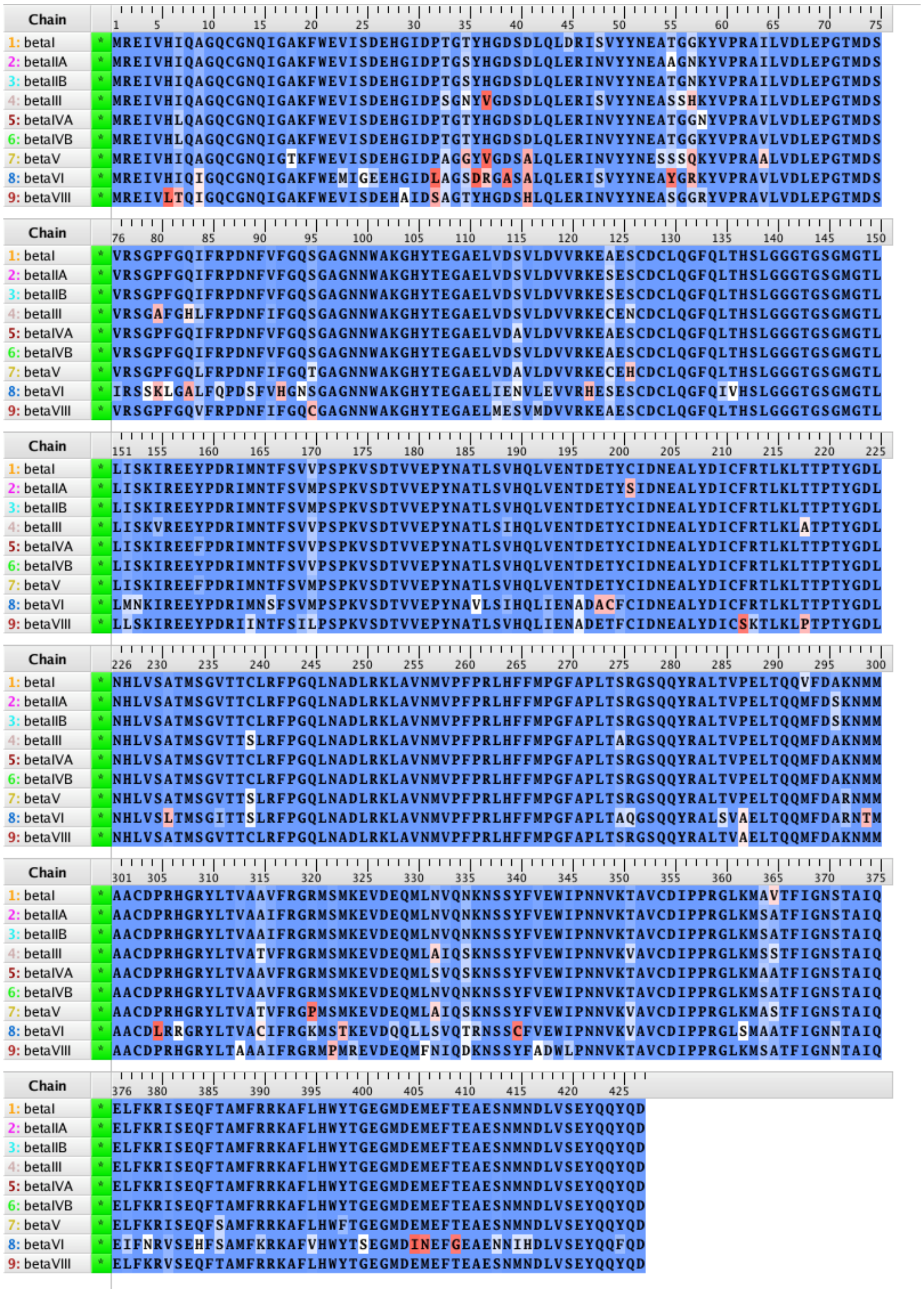


| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | CLASS | [+]Smean | [+]Smean | [+]Smean | [+]Smean | [+]Smean | [+]Smean |
| 2 | 1 | 8510 | 9871 | 10673 | 11138 | 15034 | 10920 |
| 3 | 2 | 8846 | 10177 | 11409 | 11514 | 15584 | 11181 |
| 4 | 1 | 10011 | 11171 | 12053 | 12579 | 16392 | 12306 |
| 5 | 3 | 8947 | 10486 | 11436 | 11768 | 16081 | 11580 |
| 6 | 1 | 8300 | 9751 | 10696 | 11122 | 15228 | 10802 |
| 7 | 3 | 8510 | 9871 | 10673 | 11138 | 15034 | 10920 |
| 8 | 2 | 9142 | 10421 | 11358 | 11511 | 15619 | 11231 |
| 9 | 1 | 9839 | 11268 | 11921 | 12184 | 16075 | 12029 |
| 10 | 1 | 8523 | 9891 | 10979 | 11514 | 15301 | 11100 |
| 11 | 1 | 8657 | 9797 | 10821 | 11281 | 15318 | 10984 |
| 12 | 1 | 8904 | 10325 | 11085 | 11561 | 15546 | 11470 |

**Figure 4**

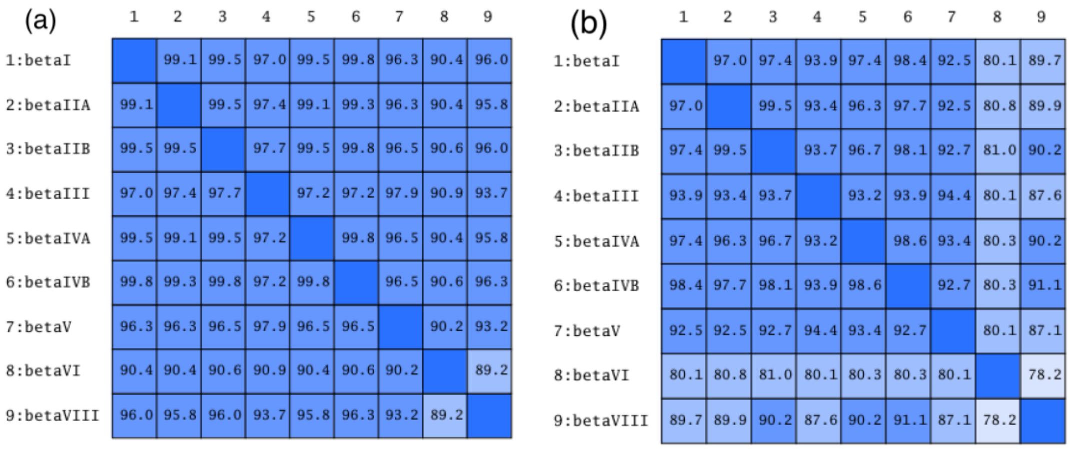


# Figure 5



```
Chain          1   5    10   15   20   25   30   35   40   45   50   55   60   65   70   75
1: betaI     * MREIVHIQAGQCGNQIGAKFWEVISDEHGIDPTGTYHGDSDLQLDRISVYYNEATGGKYVPRAILVDLEPGTMDS
2: betaIIA   * MREIVHIQAGQCGNQIGAKFWEVISDEHGIDPTGSYHGDSDLQLERINVYYNEAAGNKYVPRAILVDLEPGTMDS
3: betaIIB   * MREIVHIQAGQCGNQIGAKFWEVISDEHGIDPTGSYHGDSDLQLERINVYYNEATGNKYVPRAILVDLEPGTMDS
4: betaIII   * MREIVHIQAGQCGNQIGAKFWEVISDEHGIDPSGNYVGDSDLQLERISVYYNEASSHKYVPRAILVDLEPGTMDS
5: betaIVA   * MREIVHLQAGQCGNQIGAKFWEVISDEHGIDPTGTYHGDSDLQLERINVYYNEATGGNYVPRAVLVDLEPGTMDS
6: betaIVB   * MREIVHLQAGQCGNQIGAKFWEVISDEHGIDPTGTYHGDSDLQLERINVYYNEATGGKYVPRAVLVDLEPGTMDS
7: betaV     * MREIVHIQAGQCGNQIGTKFWEVISDEHGIDPAGGYVGDSALQLERINVYYNESSSQKYVPRAALVDLEPGTMDS
8: betaVI    * MREIVHIQIGQCGNQIGAKFWEMIGEEHGIDLAGSDRGASALQLERISVYYNEAYGRKYVPRAVLVDLEPGTMDS
9: betaVIII  * MREIVLTQIGQCGNQIGAKFWEVISDEHAIDSAGTYHGDSHLQLERINVYYNEASGGRYVPRAVLVDLEPGTMDS

Chain          76  80   85   90   95   100  105  110  115  120  125  130  135  140  145  150
1: betaI     * VRSGPFGQIFRPDNFVFGQSGAGNNWAKGHYTEGAELVDSVLDVVRKEAESCDCLQGFQLTHSLGGGTGSGMGTL
2: betaIIA   * VRSGPFGQIFRPDNFVFGQSGAGNNWAKGHYTEGAELVDSVLDVVRKESESCDCLQGFQLTHSLGGGTGSGMGTL
3: betaIIB   * VRSGPFGQIFRPDNFVFGQSGAGNNWAKGHYTEGAELVDSVLDVVRKESESCDCLQGFQLTHSLGGGTGSGMGTL
4: betaIII   * VRSGAFGHLFRPDNFIFGQSGAGNNWAKGHYTEGAELVDSVLDVVRKECENCDCLQGFQLTHSLGGGTGSGMGTL
5: betaIVA   * VRSGPFGQIFRPDNFVFGQSGAGNNWAKGHYTEGAELVDAVLDVVRKEAESCDCLQGFQLTHSLGGGTGSGMGTL
6: betaIVB   * VRSGPFGQIFRPDNFVFGQSGAGNNWAKGHYTEGAELVDSVLDVVRKEAESCDCLQGFQLTHSLGGGTGSGMGTL
7: betaV     * VRSGPFGQLFRPDNFIFGQTGAGNNWAKGHYTEGAELVDAVLDVVRKECEHCDCLQGFQLTHSLGGGTGSGMGTL
8: betaVI    * IRSSKLGALFQPDSFVHGNSGAGNNWAKGHYTEGAELIENVLEVVRHESESCDCLQGFQIVHSLGGGTGSGMGTL
9: betaVIII  * VRSGPFGQVFRPDNFIFGQCGAGNNWAKGHYTEGAELMESVMDVVRKEAESCDCLQGFQLTHSLGGGTGSGMGTL

Chain          151 155  160  165  170  175  180  185  190  195  200  205  210  215  220  225
1: betaI     * LISKIREEYPDRIMNTFSVVPSPKVSDTVVEPYNATLSVHQLVENTDETYCIDNEALYDICFRTLKLTTPTYGDL
2: betaIIA   * LISKIREEYPDRIMNTFSVMPSPKVSDTVVEPYNATLSVHQLVENTDETYSIDNEALYDICFRTLKLTTPTYGDL
3: betaIIB   * LISKIREEYPDRIMNTFSVMPSPKVSDTVVEPYNATLSVHQLVENTDETYCIDNEALYDICFRTLKLTTPTYGDL
4: betaIII   * LISKVREEYPDRIMNTFSVVPSPKVSDTVVEPYNATLSIHQLVENTDETYCIDNEALYDICFRTLKLATPTYGDL
5: betaIVA   * LISKIREEFPDRIMNTFSVVPSPKVSDTVVEPYNATLSVHQLVENTDETYCIDNEALYDICFRTLKLTTPTYGDL
6: betaIVB   * LISKIREEYPDRIMNTFSVVPSPKVSDTVVEPYNATLSVHQLVENTDETYCIDNEALYDICFRTLKLTTPTYGDL
7: betaV     * LISKIREEFPDRIMNTFSVMPSPKVSDTVVEPYNATLSVHQLVENTDETYCIDNEALYDICFRTLKLTTPTYGDL
8: betaVI    * LMNKIREEYPDRIMNSFSVMPSPKVSDTVVEPYNAVLSIHQLIENADACFCIDNEALYDICFRTLKLTTPTYGDL
9: betaVIII  * LLSKIREEYPDRIINTFSILPSPKVSDTVVEPYNATLSVHQLIENADETFCIDNEALYDICSKTLKLPTPTYGDL

Chain          226 230  235  240  245  250  255  260  265  270  275  280  285  290  295  300
1: betaI     * NHLVSATMSGVTTCLRFPGQLNADLRKLAVNMVPFPRLHFFMPGFAPLTSRGSQQYRALTVPELTQQVFDAKNMM
2: betaIIA   * NHLVSATMSGVTTCLRFPGQLNADLRKLAVNMVPFPRLHFFMPGFAPLTSRGSQQYRALTVPELTQQMFDSKNMM
3: betaIIB   * NHLVSATMSGVTTCLRFPGQLNADLRKLAVNMVPFPRLHFFMPGFAPLTSRGSQQYRALTVPELTQQMFDSKNMM
4: betaIII   * NHLVSATMSGVTTSLRFPGQLNADLRKLAVNMVPFPRLHFFMPGFAPLTARGSQQYRALTVPELTQQMFDAKNMM
5: betaIVA   * NHLVSATMSGVTTCLRFPGQLNADLRKLAVNMVPFPRLHFFMPGFAPLTSRGSQQYRALTVPELTQQMFDAKNMM
6: betaIVB   * NHLVSATMSGVTTCLRFPGQLNADLRKLAVNMVPFPRLHFFMPGFAPLTSRGSQQYRALTVPELTQQMFDAKNMM
7: betaV     * NHLVSATMSGVTTSLRFPGQLNADLRKLAVNMVPFPRLHFFMPGFAPLTSRGSQQYRALTVPELTQQMFDARNMM
8: betaVI    * NHLVSLTMSGITTSLRFPGQLNADLRKLAVNMVPFPRLHFFMPGFAPLTAQGSQQYRALSVAELTQQMFDARNTM
9: betaVIII  * NHLVSATMSGVTTCLRFPGQLNADLRKLAVNMVPFPRLHFFMPGFAPLTSRGSQQYRALTVAELTQQMFDAKNMM

Chain          301 305  310  315  320  325  330  335  340  345  350  355  360  365  370  375
1: betaI     * AACDPRHGRYLTVAAVFRGRMSMKEVDEQMLNVQNKNSSYFVEWIPNNVKTAVCDIPPRGLKMAVTFIGNSTAIQ
2: betaIIA   * AACDPRHGRYLTVAAIFRGRMSMKEVDEQMLNVQNKNSSYFVEWIPNNVKTAVCDIPPRGLKMSATFIGNSTAIQ
3: betaIIB   * AACDPRHGRYLTVAAIFRGRMSMKEVDEQMLNVQNKNSSYFVEWIPNNVKTAVCDIPPRGLKMSATFIGNSTAIQ
4: betaIII   * AACDPRHGRYLTVATVFRGRMSMKEVDEQMLAIQSKNSSYFVEWIPNNVKVAVCDIPPRGLKMSSTFIGNSTAIQ
5: betaIVA   * AACDPRHGRYLTVAAVFRGRMSMKEVDEQMLSVQSKNSSYFVEWIPNNVKTAVCDIPPRGLKMAATFIGNSTAIQ
6: betaIVB   * AACDPRHGRYLTVAAVFRGRMSMKEVDEQMLNVQNKNSSYFVEWIPNNVKTAVCDIPPRGLKMSATFIGNSTAIQ
7: betaV     * AACDPRHGRYLTVATVFRGPMSMKEVDEQMLAIQSKNSSYFVEWIPNNVKVAVCDIPPRGLKMASTFIGNSTAIQ
8: betaVI    * AACDLRRGRYLTVACIFRGKMSTKEVDQQLLSVQTRNSSCFVEWIPNNVKVAVCDIPPRGLSMAATFIGNNTAIQ
9: betaVIII  * AACDPRHGRYLTAAAIFRGRMPMREVDEQMFNIQDKNSSYFADWLPNNVKTAVCDIPPRGLKMSATFIGNNTAIQ

Chain          376 380  385  390  395  400  405  410  415  420  425
1: betaI     * ELFKRISEQFTAMFRRKAFLHWYTGEGMDEMEFTEAESNMNDLVSEYQQYQD
2: betaIIA   * ELFKRISEQFTAMFRRKAFLHWYTGEGMDEMEFTEAESNMNDLVSEYQQYQD
3: betaIIB   * ELFKRISEQFTAMFRRKAFLHWYTGEGMDEMEFTEAESNMNDLVSEYQQYQD
4: betaIII   * ELFKRISEQFTAMFRRKAFLHWYTGEGMDEMEFTEAESNMNDLVSEYQQYQD
5: betaIVA   * ELFKRISEQFTAMFRRKAFLHWYTGEGMDEMEFTEAESNMNDLVSEYQQYQD
6: betaIVB   * ELFKRISEQFTAMFRRKAFLHWYTGEGMDEMEFTEAESNMNDLVSEYQQYQD
7: betaV     * ELFKRISEQFSAMFRRKAFLHWFTGEGMDEMEFTEAESNMNDLVSEYQQYQD
8: betaVI    * EIFNRVSEHFSAMFKRKAFVHWYTSEGMDINEFGEAENNIHDLVSEYQQFQD
9: betaVIII  * ELFKRVSEQFTAMFRRKAFLHWYTGEGMDEMEFTEAESNMNDLVSEYQQYQD
```

**Figure 6**



(a)

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1:betaI | | 99.1 | 99.5 | 97.0 | 99.5 | 99.8 | 96.3 | 90.4 | 96.0 |
| 2:betaIIA | 99.1 | | 99.5 | 97.4 | 99.1 | 99.3 | 96.3 | 90.4 | 95.8 |
| 3:betaIIB | 99.5 | 99.5 | | 97.7 | 99.5 | 99.8 | 96.5 | 90.6 | 96.0 |
| 4:betaIII | 97.0 | 97.4 | 97.7 | | 97.2 | 97.2 | 97.9 | 90.9 | 93.7 |
| 5:betaIVA | 99.5 | 99.1 | 99.5 | 97.2 | | 99.8 | 96.5 | 90.4 | 95.8 |
| 6:betaIVB | 99.8 | 99.3 | 99.8 | 97.2 | 99.8 | | 96.5 | 90.6 | 96.3 |
| 7:betaV | 96.3 | 96.3 | 96.5 | 97.9 | 96.5 | 96.5 | | 90.2 | 93.2 |
| 8:betaVI | 90.4 | 90.4 | 90.6 | 90.9 | 90.4 | 90.6 | 90.2 | | 89.2 |
| 9:betaVIII | 96.0 | 95.8 | 96.0 | 93.7 | 95.8 | 96.3 | 93.2 | 89.2 | |

(b)

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1:betaI | | 97.0 | 97.4 | 93.9 | 97.4 | 98.4 | 92.5 | 80.1 | 89.7 |
| 2:betaIIA | 97.0 | | 99.5 | 93.4 | 96.3 | 97.7 | 92.5 | 80.8 | 89.9 |
| 3:betaIIB | 97.4 | 99.5 | | 93.7 | 96.7 | 98.1 | 92.7 | 81.0 | 90.2 |
| 4:betaIII | 93.9 | 93.4 | 93.7 | | 93.2 | 93.9 | 94.4 | 80.1 | 87.6 |
| 5:betaIVA | 97.4 | 96.3 | 96.7 | 93.2 | | 98.6 | 93.4 | 80.3 | 90.2 |
| 6:betaIVB | 98.4 | 97.7 | 98.1 | 93.9 | 98.6 | | 92.7 | 80.3 | 91.1 |
| 7:betaV | 92.5 | 92.5 | 92.7 | 94.4 | 93.4 | 92.7 | | 80.1 | 87.1 |
| 8:betaVI | 80.1 | 80.8 | 81.0 | 80.1 | 80.3 | 80.3 | 80.1 | | 78.2 |
| 9:betaVIII | 89.7 | 89.9 | 90.2 | 87.6 | 90.2 | 91.1 | 87.1 | 78.2 | |

**Figure 7**

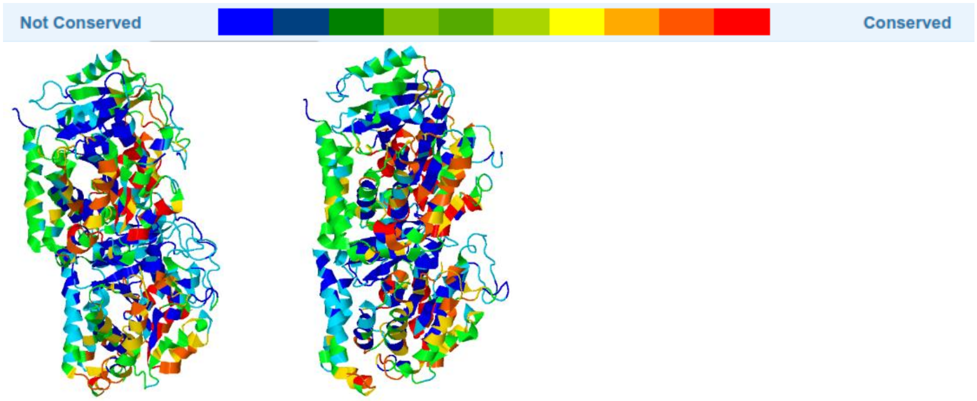


**Figure 8**

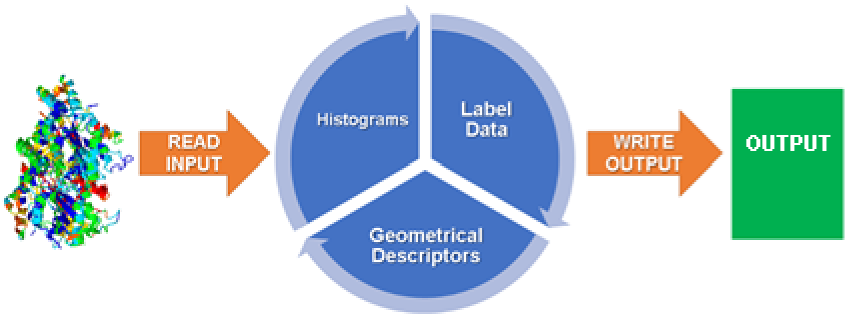


**Figure 9**

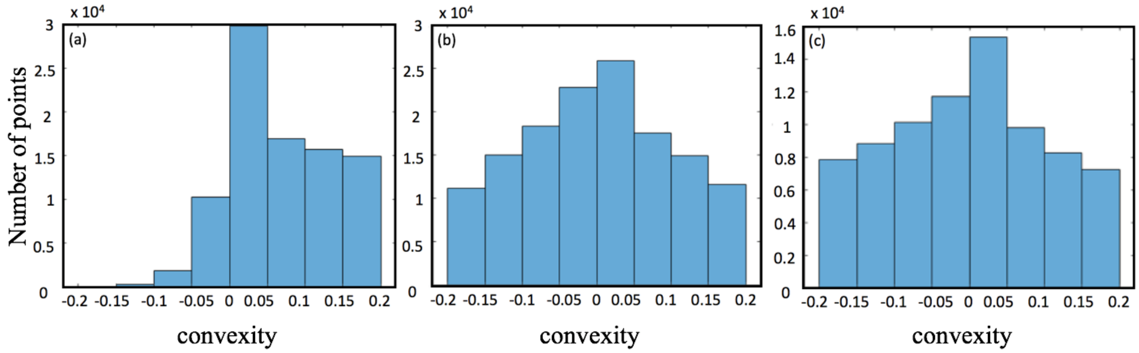
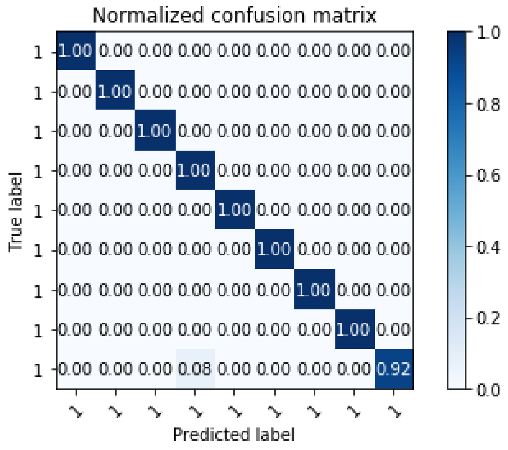


**Figure 10.**



**Tables' captions:**

**Table 1:** Numbers of tubulin isotype structures used.

**Table 2:** Sample numbers in the binary classification between tubulin and FtsZ.

**Table 3:** Number of Tubulin isotypes used.

**Table 4:** Tubulin isotypes accuracy results.

**Table 5:** Accuracy results for the tubulin and FtsZ binary classification.

**Table 6:** 2R6R, 2VAM, 2VAP and 2VAM samples.

**Table 7:** 2R6R, 2VAM, 2VAP and 2VAM experiment.

**Tables:**

**Table 1**

| **Isotypes** | Beta I | Beta IIa | Beta IIb | Beta III | Beta IVa | Beta IVb | Beta V | Beta VI | Beta VIII |
|---|---|---|---|---|---|---|---|---|---|
| **Samples** | 123 | 128 | 94 | 57 | 128 | 68 | 107 | 62 | 125 |

**Table 2**

| **Protein** | **Samples** |
|---|---|
| Tubulin | 112 |
| FtsZ | 65 |

**Table 3**

| **Isotypes** | Beta I | Beta IIa | Beta IIb | Beta III | Beta IVa | Beta IVb | Beta V | Beta VI | Beta VIII |
|---|---|---|---|---|---|---|---|---|---|
| **Samples** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

**Table 4**

| **Classifier** | **Accuracy** |
|---|---|
| SVM with RBF kernel | 96.5 % |
| SVM with linear kernel | 92.4 % |
| k-means | 76.6 % |

**Table 5**

| **Classifier** | **Accuracy** |
|---|---|
| SVM with RBF kernel | 98.2 % |
| SVM with linear kernel | 97.0 % |
| k-means | 72.3 % |

**Table 6**

| Proteins | 2R6R | 2VAW | 2VAP | 2VAM |
|---|---|---|---|---|
| Samples | 175 | 170 | 168 | 170 |

**Table 7**

| Classifier | Accuracy |
|---|---|
| SVM with RBF kernel | 97.1 % |
| SVM with linear kernel | 98.0 % |
| k-means | 62.3 % |