SHREC 2025: Protein Shape Classification Methods proposed by IMATI group

Marco Guerra[1], Giorgio Palmieri[1], Andrea Ranieri, Ulderico Fugacci[1], Silvia Biasotti[1]

[1] CNR-IMATI Genova, Italy
{name.surname}@ge.imati.cnr.it

We present three different methods to tackle the proposed challenge. Then, we define a scoring strategy to combine them in a fourth one.

1 Method 1: Topological Descriptors

We consider each mesh as a topological object in $\mathbb{R}^3$, either as a point cloud or as a triangulated surface.

Alpha descriptors We compute the Alpha filtration ([1]) of the point cloud in $\mathbb{R}^3$, and its persistent homology in dimension $k = 0, 1, 2$. The resulting k-persistence diagrams are then thresholded at longer lifespans, and for each k a 5x5 persistence image is used for vectorization, resulting in a feature vector of length 75.

Radial descriptors The centroid of the point cloud is computed. A reference sphere is subdivided into 8 equal spherical sectors. We consider the intersection of the surface with each sector; for each, we compute the [.25, .5, .75] quantiles of the distribution of distance of points from the centroid; the cumulative radial potential at those quantiles; the number of persistent homology classes whose lifespan is at least one tenth of the median; the birth and death time of the longest finite H_0 bar, and the potentials of the generating vertices in the lower-star filtration. This results in 96 features.

Data preparation The descriptors are combined for each protein. We enforce rotational invariance of the classifier by augmenting each data point by the action of the group of rotations onto the radial descriptors. Given our subdivision in 8 sectors, we obtain 8 different descriptions for each protein. The descriptors are decorrelated and scaled. Finally, the dataset is

partially balanced by introducing duplicates of the more infrequent classes, and divided into 80-20 training and validation subsets.

Classification We use a small neural network with 4 fully-connected layers, whose sizes decrease linearly from the input to output size, with ReLU activations. Each layer employs dropout with $p = .25$, and we further enforce L^2-regularization.
By using data augmentation on the test set as well, we obtain a set of 8 predictions for each protein to classify. The result is then given by majority, with ties broken by a-priori class frequency.

1.1 Hardware

Processor: Intel(R) Core(TM) i9-9900KF CPU @ 3.60GHz, 64bits
Memory: 32GiB
GPU: NVIDIA Corporation TU117 [GeForce GTX 1650], with CUDA enabled

1.2 Computation Time

Computation of **descriptors** for training and test set: around 3 hours in total. Training of the **classifier**: around 30 minutes for 2000 epochs.

2 Method 2: PointNet Architecture

2.1 Dataset Construction and Preprocessing

For dataset construction, we utilized protein point clouds subjected to minimal data augmentation. Applied transformations included random spatial translation and additive Gaussian noise (empirical observations indicated that protein rotation adversely impacted performance). All point clouds were normalized to fit within a unit sphere and processed using a random sampling strategy to ensure uniform point density.

2.2 Model Architecture and Training Protocol

We employed a standard PointNet architecture (26M parameters) initialized from scratch. Pretraining experiments revealed no significant improvement in training or validation loss trajectories, with final accuracy remaining comparable to models trained without pretraining (see ablation study in Supplementary Materials).

For the final evaluation, the model was trained on the full training set for 100 epochs. To mitigate class imbalance, we applied a class-balancing strategy: samples from underrepresented classes were duplicated by a factor

proportional to the ratio of the largest class size, capped at a maximum multiplier of 10. This approach preserved dataset diversity while reducing bias toward dominant classes.

Experiments were conducted on a workstation equipped with three *NVIDIA RTX A6000 GPUs* (48GB VRAM each), an *AMD Ryzen Threadripper PRO 7965WX CPU* (24 cores/48 threads), and 128GB DDR5 RAM. The training used just one of the three GPUs.

To obtain the final prediction, we employed three PointNet models at three distinct stages of the training process: at epoch 80, at the conclusion of the training phase, and at the point of minimal validation loss. By integrating the three distinct logit vectors generated at these stages and aggregating them via summation, we derived the ultimate prediction.

3 Method 3: A Multi-View Image-Based Approach using a ViT Encoder

For Submission 3, we developed a novel image-based pipeline for protein structure classification. Starting from '.vtk' files, we implemented a PyVista-based script to generate 1024×1024-pixel RGB images. Each image comprises an 8×8 grid of 128×128-pixel sub-views, captured at 45-degree intervals along both azimuth and elevation angles to comprehensively represent the protein's 3D structure.

Dataset Curation and Splitting The raw dataset was soft-balanced to mitigate its severe class imbalance, capping duplication for underrepresented classes at 1000× and generating a final corpus of 50,000 samples. The dataset was partitioned into training and validation sets using an 80/20 random split to preserve class distribution.

Model Architecture and Training We trained a classification model using a *vit_giant_patch14_reg4_dinov2* encoder, implemented via a PyTorch and Fast.ai pipeline inside Jupyter notebook. To enhance generalization, a custom data augmentation strategy was applied:

- *Stochastic Sub-Image Rotation*: each 128×128 sub-view in the 8×8 grid was rotated by a random angle (0–359°), with the composite image reconstructed post-rotation.
- *RGBShift Augmentation*: targeted shifts were applied to the red and green channels, reflecting their dominance in the PyVista-rendered protein visualizations.
- *Resolution Scaling*: images were resized to 518×518 pixels to match the encoder's patch dimensions.

Hardware and Training Details Experiments were conducted on a workstation equipped with three *NVIDIA RTX A6000 GPUs* (48GB VRAM

each), an *AMD Ryzen Threadripper PRO 7965WX CPU* (24 cores/48 threads), and 128GB DDR5 RAM. Training utilized two GPUs, completing 10 epochs in 20 hours ($\sim$ 2 hours/epoch). Inference times averaged 2 seconds per protein screenshot generation (test set) and 150ms per forward pass using the trained model.

This approach aims at demonstrating the efficacy of multi-view representation learning for 3D biological structures, hopefully achieving robust performance while addressing both dataset complexity and class imbalance challenges.

4 Method 4: Combining different approaches

For our fourth submission, we choose a combined classification obtained by a weighted majority vote among the three methods previously described. Specifically, for each protein, we have three proposed classes each with a confidence score. In case a majority exists, we choose that class. If no majority exists, our algorithm excludes the class with lower confidence and keeps the class with a-priori highest frequency (based on the training set).

References

[1] Herbert Edelsbrunner and Ernst P. Mücke. Three-dimensional alpha shapes. *ACM Trans. Graph.*, 13(1):43–72, jan 1994.