From Unsupervised Multi-Instance Learning to Identification of Near-Native Protein Structures*

Fardina Fathmiul Alam[1] and Amarda Shehu[1,2,3,4,†]

[1] Department of Computer Science, George Mason University, Fairfax, VA
[2] Center for Advancement of Human-Machine Partnerships, George Mason University, Fairfax, VA
[3] Department of Bioengineering, George Mason University, Fairfax, VA
[4] School of Systems Biology, George Mason University, Manassas, VA
amarda@gmu.edu

Abstract

A major challenge in computational biology regards recognizing one or more biologically-active/native tertiary protein structures among thousands of physically-realistic structures generated via template-free protein structure prediction algorithms. Clustering structures based on structural similarity remains a popular approach. However, clustering organizes structures into groups and does not directly provide a mechanism to select individual structures for prediction. In this paper, we provide a few algorithms for this selection problem. We approach the problem under unsupervised multi-instance learning and address it in three stages, first organizing structures into bags, identifying relevant bags, and then drawing individual structures/instances from these bags. We present both non-parametric and parametric algorithms for drawing individual instances. In the latter, parameters are trained over training data and evaluated over testing data via rigorous metrics.

1 Introduction

The three-dimensional (tertiary) structure in which the chain of amino acids comprising a protein molecule folds in three dimensions determines to a great extent a protein's biological activities [6]. For millions of known protein sequences with no known functional or structural characterization, structure determination is key to obtain information on potential activities in the cell. Template-free protein structure prediction algorithms approach the problem of tertiary structure determination as an optimization problem. Instantiated from a given amino-acid sequence, they generate many physically-realistic tertiary structures driven the objective of minimizing a potential energy function that sums up interatomic interactions [15]. Many low-energy structures are generated. Somewhere among them hide the ones that are populated by a protein molecule under physiological conditions, also referred to as native or near-native.

*This work is supported in part by NSF IIS Grant No. 1763233, NSF FET Grant No. 1900061, NSF DMS Grant No. 1821154, and a Jeffress Trust Award.
†Corresponding Author

Recognizing the near-native structure(s) among the decoy structures remains an open problem in computational biology [12]. At low energies, energy values are not discriminative enough to point to the near-native structures. Therefore, clustering structures based on structural similarity remains a popular approach [2]. Research on effective clustering remains active [5, 19]. However, clustering organizes structures into groups and does not directly provide a mechanism to select individual structures for prediction. In this paper, we focus on such mechanisms and provide a few algorithms to address this selection problem.

In this paper, we approach the problem of identifying a near-native structure by utilizing unsupervised multi-instance learning (MIL) [9]. We realize that the popular clustering approaches carry out a learning process of building groups or bags in which (ideally) similar instances are put together, and less similar instances are separated. Building over this, we propose to proceed in three stages, first organizing structures into bags (via clustering), identifying relevant bags (clusters), and then drawing individual structures/instances from the identified bags.

Our focus is not on designing better clustering algorithms for molecular structures, though we employ here a state-of-the-art clustering algorithm that can handle non-uniform clusters, such as Gaussian Mixture Model (GMM) clustering. So as not to be handicapped by the high dimensionality of protein tertiary structures, the GMM algorithm is applied to featurized data obtained via nonlinear autoencoders. The latter are shown to provide superior dimensionality reduction over linear and other non-linear models [4]. While our focus is not on improving clustering or dimensionality reduction, in this paper we demonstrate via precise machine learning metrics that the obtained clusters are bags of high quality and warrant leveraging for single-instance/decoy selection.

Specifically, we present both non-parametric and parametric algorithms for single-instance selection. These algorithms operate over a bag/cluster predicted to be of high quality via ranking-based selection. In the proposed parametric algorithms, parameters are trained over training data and evaluated over testing data. The evaluation is carried out over several datasets containing Rosetta-generated decoys of proteins of diverse lengths and folds, employing rigorous machine learning performance metrics.

2 Method

In the interest of clarity, we first provide a summary of the proposed approach. The input is a set of the tertiary structures of a given protein molecule. These structures are generated via the Rosetta AbInitio protocol [10], which is publicly available to researchers. Each structure is specified in terms of the Cartesian $\{x, y, z\}$ coordinates of its atoms. The approach consists of three main components: organizing given structures into bags, identifying/predicting a high-quality bag (or more), and selecting a single instance/decoy from the identified bag(s). An important component of all unsupervised MIL approaches that utilize clustering is featurization of the given instances, as clustering is known to perform badly on high-dimensional data [16]. Therefore, the tertiary structures are first encoded in a latent feature space prior to being subjected to clustering. We now detail the featurization, clustering, identification of relevant clusters, and single-decoy selection algorithms utilized in the proposed methodology.

2.1 Data Featurization

If we treat each structure as a data point, the data reside in a space of thousands of dimensions, as a small-to-medium protein molecule may contain thousands of atoms. Therefore, it is imperative to reduce the dimensionality of the space (that is, featurize the data) prior to clustering.

However, the focus of our paper is not on feature design. Instead, we prefer to employ domain-agnostic, non-linear techniques demonstrated to be useful at reducing the dimensionality of molecular structure data.

Specifically, we leverage recent work in [4], which evaluates various linear and non-linear, shallow and deep autoencoders for dimensionality reduction of tertiary protein structures. The interested reader can find more information in Ref. [5]. For the purpose of the study carried out in this paper, we note that a non-linear (with parametric rectified linear unit as activation function in both the encoder and decoder), deep autoencoder is employed to map Rosetta-generated tertiary structures of a protein onto a two-dimensional (2D) feature space; that is, each structure is represented with 2 features.

2.2 Clustering

The thus-featurized structures are fed to a clustering algorithm. As we relate in Section 1, the focus of this paper is not on designing a novel clustering algorithm; nor is it on evaluating different clustering algorithms, though this line of research can certainly be investigated in future work. Instead, we focus on making connections between the problem of single-decoy selection with unsupervised MIL. In summary, in MIL, each object is represented by a bag composed of multiple instances instead of by a single instance as in the traditional learning setting. The majority of works focus on the multi-instance prediction problem, where each bag is associated with a binary (classification) or real-valued (regression) label. However, tertiary protein structures do not come with labels and instead impose an unsupervised MIL setting. This setting has also been studied and typically via clustering algorithms [9]. Under the umbrella of unsupervised MIL, the problem of single-decoy selection can be approached in three steps, first organizing structures into bags or clusters, then devising a mechanism to focus on one or more bags likely to be relevant for single-decoy selection, and then applying an algorithm that is likely to draw a near-native structure from the identified bag(s).

To obtain the bags containing multiple instances (near-native and non-native structures), we elect to choose a popular, yet sophisticated clustering algorithm that can handle non-uniform clusters, such as the Gaussian Mixture Modeling (GMM) [8]. In summary, the GMM clustering algorithm implements the expectation-maximization algorithm for fitting mixture-of-Gaussian models. To perform model selection in GMM, which concerns choosing both the covariance type and the number of components/clusters, we use information-theoretic criteria, such as the Bayesian Information criterion (BIC) [14] and the Akaike Information criterion (AIC) [1]. The number of components and the type of covariance (full, spherical, tied, or diagonal) are determined by minimizing the AIC/BIC ratio. Model selection for GMM is provided in the sklearn.mixture Python library.

An additional reason we prefer GMM over other clustering algorithms is due to the multi-cluster membership feature that is a rich setting we exploit in this paper to present and evaluate a variety of single-decoy selection algorithms. In GMM, a data point is not uniquely assigned to one cluster. Instead, GMM associates each data point with a probability distribution of cluster membership, where probabilities are provided for the points to belong to any of the clusters identified (with the total adding up to 1). So, the clustering component of our methodology provides two types of information: (1) the clusters identified over the data points, and (2) a vector of membership probability per cluster for each data point.

2.3 Ranking-based Cluster Selection and Evaluation

An important question for single-decoy selection is how to leverage the obtained clusters. The answer to this question needs to consider the quality of clusters obtained. Since one does *not a priori* know how near-native structures are distributed among the clusters identified, a reasonable way to proceed is by considering the characteristics of clusters. We proceed with a rather straightforward characteristic, such as size. That is, the hypothesis is that the largest cluster is more likely to contain near-native structures than other clusters.

In Section 3, we evaluate this hypothesis by measuring the *purity* of the selected (largest) cluster. This metric has been introduced by us in related work on clustering algorithms [3] and is related to precision. Specifically, purity measures the fraction of the number of near-native decoys in a cluster over the size (total number of decoys) of the cluster. The determination on whether a decoy is near-native or not is based on a threshold over the popular least root-mean-squared-deviation (lRMSD) metric [11]. The latter first finds an optimal superimposition of a decoy to a known native structure to remove differences due to translation and rotation in 3D and then averages the Euclidean distance over the number of dimensions/features.

Other characteristics can be investigated in future work, but in this paper we effectuate a decision so as to proceed and evaluate single-decoy selection over an identified cluster. That is, the output of this ranking-based selection that gets fed as a input to a single-decoy selection algorithm is the largest cluster (with the decoys that reside in it) and the vector of probability-per-cluster for each of the decoys in the largest cluster. It is important to note that the GMM algorithm assigns a decoy to the cluster for which the probability-per-cluster of the decoy is largest. By additionally considering the probabilities for a decoy to belong to other clusters, we obtain a broader picture that we leverage for single-decoy selection as described below.

2.4 Single-Decoy Selection

We present two single-decoy selection algorithms, non-parametric and parametric, as describe below. Before relating details, we note that we evaluate these algorithms by evaluating the quality of a decoy they select from a given bag of decoys. We do so in terms of loss. Specifically, we measure the lRMSD of the best decoy we could have selected and compare how much larger the lRMSD of the actual-selected decoy is to a given native structure. Obviously, we can only do so on a benchmark dataset where we do not operate in a blind setting, but know the native structure for the purpose of evaluation. Specifically, let us refer to the lRMSD of the best decoy (closest to the native structure) to the known native structure as *lRMSD(Best Decoy)*. Loss is then measured as *lRMSD(Selected Decoy)* − *lRMSD(Best Decoy)*. The smaller the value, the better the performance of the algorithm. The single-decoy selection algorithms we present draw from a given bag or bags – cluster(s) – in a non-deterministic/probabilistic manner. Therefore, we measure their average loss over L independent drawings (with replacement). In our evaluations, we employ $L = \{5, 10\}$.

The non-parametric algorithm is utilized as a baseline in our evaluation. It is a naive algorithm that draws a decoy at random from a given set. The parametric algorithm is more sophisticated, as it utilizes the per-cluster-membership probabilities of each decoy in a given set of decoys. Note that there is no need to associate this set with the largest cluster/bag. The algorithms are agnostic to how the set was obtained. What the parametric algorithm relies on is the expectation that each decoy is associated with a vector of values in $[0, 1]$, where a value at index i encodes the probability that the decoy belongs to bag i. The order of bags (and corresponding indices) relates to size (in the above ranking-based selection, clusters are ordered by size, from largest to smallest).

For each decoy, the parametric algorithm finds the probability that it belongs to the largest bag and measures the range of these probabilities over all decoys in the set provided to it. The intuition we leverage is that this probability potentially provides us with insight into the quality of a decoy. We hypothesize that the larger the probability that a decoy belongs to the largest bag, the more likely it is to be of high quality (near-native in our case).

After obtaining the range of probabilities (belong to the largest bag) over the decoys, the parametric algorithm proceeds by removing/discarding from the set decoys of low quality; that is, decoys where this probability is very low, are discarded. The algorithm makes use of a threshold parameter τ for this purpose, and grid search over a training dataset is carried out to determine a reasonable value for τ. This value is then later utilized in evaluating the parametric single-decoy selection algorithm over a testing dataset (in the context of the overall performance in terms of loss). Only decoys whose probability of belonging to the largest cluster is no lower than τ are retained in the set. Random drawing is utilized over the reduced set that is now expected to contain higher-quality decoys.

2.5 Implementation Details

The algorithms presented here are implemented in Python. We make use of Python's sklearn library for the GMM implementation and model search. The autoencoder-based featurization, which is based on our earlier work, is implemented via Keras [7]. Experiments are carried out on the Mason Argo supercomputing cluster where we used Dell Compute nodes with 16 to 28 cores (Dual Intel Xeon CPUs) and 64GB to 1.5TB RAM memory per core. Clustering the data (including model search) takes anywhere from 0.5s to 2hrs, depending on dataset size.

3 Results

3.1 Datasets

We employ a benchmark dataset of 18 proteins of varying lengths (ranging from 53 to 123 amino acids) and folds (α, β, and $\alpha + \beta$) that are used widely for evaluation [17, 18], shown in Table 1. We experiment with 18 proteins of different lengths and folds. These proteins constitute a benchmark dataset often used by decoy generation algorithms [13, 18]. We used the Rosetta *template-free* (decoy generation) protocol [10] to generate $51,000$ to $68,000$ decoys per target. For each decoy, we only retain its all-atom Cartesian coordinates. The energy of each decoy is measured via the Rosetta all-atom internal energy function (score12) measured in Rosetta Energy Units (REUs).

Table 1 presents all the 18 proteins arranged into 3 different categories/levels of difficulty (easy, medium, and hard). These levels have been determined using the minimum lRMSD between the generated decoys and a known native structure of the corresponding protein (obtained from the PDB). The size of the decoy ensemble $|\Omega|$ for each target is shown in Column 5. The proteins in this dataset are identified via the PDB entry id of a known native structure for them. The 4-letter PDB ids are shown in Column 2; the fifth letter identifies the chain in a multi-chain PDB entry. Column 7, which shows the percentage of near-native decoys (within dist_threshold of the known native structure), conveys the extreme imbalance of the decoy datasets; in some cases, the near-native decoys constitute less than 5% of the dataset.

Table 1: Testing dataset (* denotes proteins with a predominant β fold and a short helix). The chain extracted from a multi-chain PDB entry (shown in Column 2) to be used as the native structure is shown in parentheses. The fold of the known native structure is shown in Column 3. The length of the protein sequence (#aas) is shown in Column 4. The size of the Rosetta-generated decoy dataset is shown in Column 5. Column 6 shows the minimum lRMSD over decoys from the known native structure. Column 7 shows the percentage of near-native decoys (within dist_threshold of the known native structure).

Difficulty	**PDB id**	**Fold**	**# aas**	**# decoys**	**min lRMSD** (Å)	**% near-native**
Easy	1ail	α	70	58,491	0.50	6.352
	1dtd(B)	$\alpha+\beta$	61	58,745	0.51	22.827
	1wap(A)	β	68	68,000	0.60	10.192
	1tig	$\alpha+\beta$	88	60,000	0.60	15.109
	1dtj(A)	$\alpha+\beta$	74	60,500	0.68	22.435
Medium	1hz6(A)	$\alpha+\beta$	64	60,000	0.72	11.325
	1c8c(A)	β^*	64	65,000	1.08	10.882
	2ci2	$\alpha+\beta$	65	60,000	1.21	22.443
	1bq9	β	53	61,000	1.30	1.565
	1hhp	β^*	99	60,000	1.52	2.486
	1fwp	$\alpha+\beta$	69	51,724	1.56	5.819
	1sap	β	66	66,000	1.75	2.304
Hard	2h5n(D)	α	123	54,795	2.00	0.845
	2ezk	α	93	54,626	2.56	13.047
	1aoy	α	78	57,000	3.26	10.923
	1cc5	α	83	55,000	3.95	5.529
	1isu(A)	*coil*	62	60,000	5.53	5.304
	1aly	β	146	53,000	8.53	2.779

3.2 Experimental Setup

We proceed to show three sets of results. First, we carry out a ranking-based analysis to observe where the highest-purity cluster (obtained via GMM) falls when ranking clusters based on different criteria. We then relate the quality of the largest cluster in terms of its purity and the largest cluster after the parametric single-decoy selection algorithm removes decoys deemed to be of low quality. We then show the ranking of the highest-quality cluster (measured via purity) in different orderings of the cluster based on different criteria. Finally, we relate the average loss (over $L \in 5, 10$ drawings) obtained by the non-parametric (baseline) and parametric single-decoy selection algorithms operating over the largest cluster.

We note that the parametric single-decoy selection algorithm depends on the threshold parameter τ. We have carried out grid search over different parameter values in a subset of the datasets. We refer to the latter as the training dataset, as we learn a reasonable threshold value on a subset of the datasets and then later relate results on application of the threshold on the rest of the datasets. Specifically, we select proteins from each category (for a total of 6 datasets) over which to learn the optimal threshold parameter value. These are 1dtj(A), 1dtd(B), 1bq9, 1ail, 1aoy, and 1cc5. Table 2 relates the average loss (over $L = 5$ and $L = 10$ drawings) obtained at the different thresholds and shows that albeit variations, $\tau = 80\%$ is a reasonable threshold. When $\tau = 80\%$, the lowest loss is obtained for $L = 5$ on 4/6 of the datasets (1dtd(B), 1dtj(A),

1aoy, and 1cc5). When $L = 10$, the lowest loss is obtained for $\tau = 80\%$ on 2/6 of the datasetss (1dtj(A) and 1aoy); on the two datasets of 1dtd(B) and 1cc5, the loss is similar to the lowest loss achieved on these datasets.

Table 2: Average Loss over Different Thresholds. Entries in Columns 3-8 are shown as pairs, relating the average loss over $L = 5$ and the average loss over $L = 10$.

Difficulty	PDB id	$\tau = 51\%$	$\tau = 60\%$	$\tau = 70\%$	$\tau = 80\%$	$\tau = 90\%$
Easy	1ail	1.680, 1.492	1.800, 2.766	1.755, 1.637	1.740, 1.723	1.662, 1.296
	1dtd(B)	4.320, 5.830	2.843, 3.258	3.886, 3.612	3.445, 3.433	3.731, 4.245
	1dtj(A)	0.465, 1.395	0.813, 1.281	1.277, 1.084	0.345, 1.179	1.920, 1.248
Medium	1bq9	3.201, 5.441	4.639, 4.504	3.476, 3.466	6.050, 3.931	4.151, 6.018
Hard	1aoy	6.146, 5.403	5.120, 5.892	4.770, 6.317	4.473, 5.203	6.068, 5.666
	1cc5	8.046, 6.120	6.839, 6.830	7.817, 7.968	6.492, 6.136	6.547, 7.296

3.3 Quality of Selected Cluster

We first track the rank of the highest-purity cluster in various orderings based on criteria, such as size (largest to smallest) and average lRMSD over decoys in a cluster (lowest to highest). Table 3 shows that for most of the datasets (particularly those in the Easy and Medium categories), the ranks are low. This is an encouraging result, as it suggests that selecting based on size is likely to yield purer clusters.

Table 3: Ranks of the highest-purity cluster, when clusters are ordered based on different criteria. In Column 1, clusters are ranked by size (largest to smallest). In Column 2, they are ordered by the average lRMSD over cluster decoys (from lowest to largest average lRMSD).

Difficulty	**PDB id**	**Rank (order by size)**	**Rank (order by average lRMSD)**
Easy	1ail	1	1
	1dtd(B)	2	2
	1wap(A)	1	1
	1tig	1	1
	1dtj(A)	1	1
Medium	1hz6(A)	2	2
	1c8c(A)	2	2
	2ci2	1	1
	1bq9	1	1
	1hhp	3	3
	1fwp	1	1
	1sap	1	1
Hard	2h5n(D)	4	4
	2ezk	4	4
	1aoy	3	3
	1cc5	2	2
	1isu(A)	7	7
	1aly	1	1

Figure 1 shows (in the red bars) the purity of the largest cluster obtained by GMM over each dataset. The datasets are identified based on the PDB id of the corresponding known native structure. As expected, the results show that the datasets in the Easy and Medium categories have overall a higher purity (of their largest cluster) over the datasets in the Hard category. This is expected, as the datasets in the Hard category contain very few near-natives (see Column 7 in Table 1. This result additionally informs on the fact that single-decoy selection is expected to be challenging.

The blue bars in Figure 1 show the purity recalculated over the remaining decoys in the largest cluster, after the parametric single-decoy selection algorithm removes decoys determined to be of low quality (as based on their per-cluster membership probabilities). Overall, the blue bars are higher than the red bars, indicating that the purity increases after the parametric algorithm removes decoys that do not meet the threshold. This result indirectly confirms the hypothesis that the parametric algorithm is more likely to remove non-native over near-native decoys. As, such the application of the parametric algorithm for single-decoy selection is now further warranted, as the algorithm now draws at random over a subset of decoys that has higher purity than the original set (the largest cluster). As such, this algorithm is expected to outperform drawing at random over the largest cluster, which is what the non-parametric, baseline algorithm carries out. Fig. 2, which we relate next, confirms this observation.

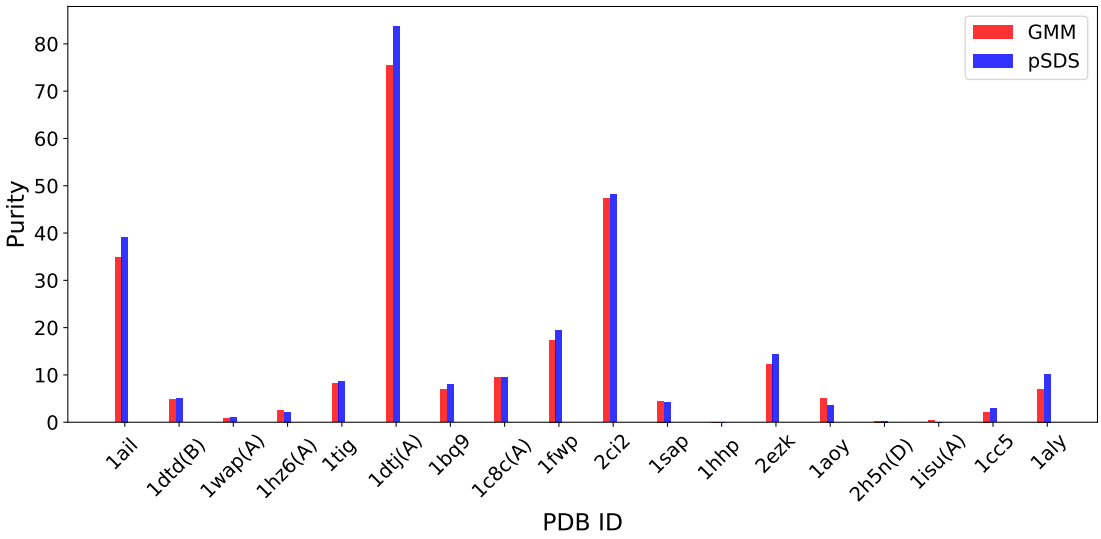

Figure 1: The purity of the largest cluster obtained via GMM and pSDS over each dataset. Two settings are shown, comparing the quality of clustering GMM, input space over the quality of clustering parametric algorithms pSDS.

3.4 Loss-based Analysis

Figure 2 compares the average loss (over $L = 5$ and $L = 10$) of the non-parametric and parametric single-decoy selection algorithms. For ease of presentation, we refer to these algorithms as npSDS and pSDS, respectively, in Figure 2. The results in Figure 2 show clearly that the parametric algorithm (pSDS) achieves, on average, lower loss than the non-parametric one.

This confirms that decoys with lower probability to belong to the largest cluster are of lower quality and removing them facilitates drawing a decoy that is more likely to be near-native.

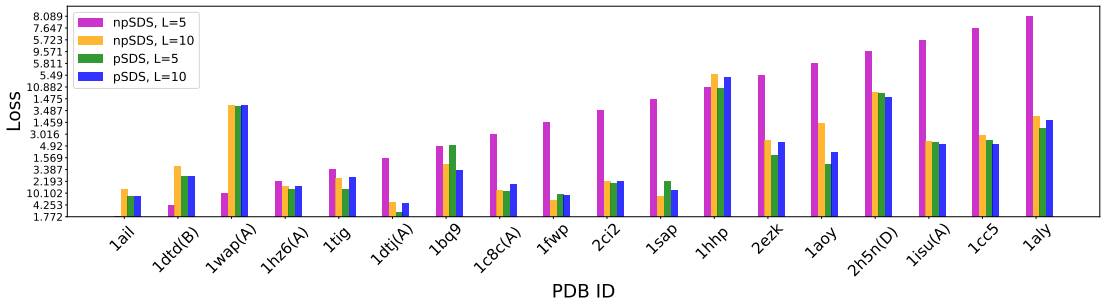

Figure 2: Comparison of average loss (over L decoys drawn with replacement) over the non-parametric (npSDS) and parametric (pSDS) decoy selection algorithms. Loss values are shown on the y-axis. The x-axis lists the various datasets, identifying them by the PDB id of the corresponding known native structure.

4 Conclusion

In this paper, we have developed an approach that leverages unsupervised multi-instance learning for instance search/prediction, inspired by a central problem in computational biology. Specifically, we have leveraged soft clustering via the GMM algorithm to investigate non-parametric and parametric algorithms for selecting a near-native structure over decoy structures generated for a given protein sequence via template-free protein structure prediction algorithms.

The presented methodology utilizes featurization, clustering, and ranking-based selection of clusters. While our focus in this paper has been on single-decoy selection, advances in featurization, clustering, and prediction of high-quality clusters present interesting directions for further research. It is worth noting that in the evaluation presented here, we have focused on 2D featurizations of protein tertiary structures. Based on a preliminary evaluation, we expect similar results when considering the 5D-10D regime (data not shown) but anticipate a worsening of performance on higher dimensionalities due to the challenge that high dimensionalities present to clustering algorithms [16]. In addition, we plan to investigate additional single-decoy selection algorithms that do not rely on parameters but still leverage soft clustering.

5 Acknowledgements

Computations were run on ARGO, a research computing cluster provided by the Office of Research Computing at George Mason University.

References

[1] K. Aho, D. Derryberry, and T. Peterson. Model selection for ecologists: the worldviews of AIC and BIC. *Ecology*, 95(3):631–636, 2014.

[2] N. Akhter, L. Hassan, Z. Rajabi, D. Barbará, and A. Shehu. Learning organizations of protein energy landscapes: An application on decoy selection in template-free protein structure prediction. In A. Kister, editor, *Protein Supersecondary Structure*, Methods in Molecular Biology. Springer, Fairfax, VA, 2018.

[3] N. Akhter and A. Shehu. From extraction of local structures of protein energy landscapes to improved decoy selection in template-free protein structure prediction. *Molecules*, 23(1):216, 2018.

[4] F. F. Alam, T. Rahman, and A. Shehu. Learning reduced latent representations of protein structure data. In *Intl Conf on Bioinf and Biomed Workshops (BIBMW): Comput Struct Biol Workshop (CSBW)*, pages 1–6, Niagara Falls, NY, September 2019. ACM.

[5] A. Alapati and D. Bhattacharya. clustq: Efficient protein decoy clustering using superposition-free weighted internal distance comparisons. In *Conf on Bioinf and Comput Biol (BCB)*, pages 307–314. ACM, 2018.

[6] D. D. Boehr and P. E. Wright. How do proteins interact? *Science*, 320(5882):1429–1430, 2008.

[7] François Chollet et al. Keras. https://keras.io, 2015.

[8] M. R. Gupta and Y. Chen. Theory and use of the em algorithm. *Foundations and Trends in Signal Processing*, 4(3):223–296, 2010.

[9] F. Herrera, S. Ventura, R. Bello, C. Cornelis, A. Zafra, D. Sánchez-Tarragó, and S. Vluymans. Unsupervised multiple instance learning. In *Multiple Instance Learning*, page 141–167. Springer, 2016.

[10] A. Leaver-Fay, M. Tyka, S. M. Lewis, O. F. Lange, J. Thompson, R. Jacak, et al. ROSETTA3: an object-oriented software suite for the simulation and design of macromolecules. *Methods Enzymol*, 487:545–574, 2011.

[11] A. D. McLachlan. A mathematical procedure for superimposing atomic coordinates of proteins. *Acta Cryst A*, 26(6):656–657, 1972.

[12] R. Nussinov, C.-J. Tsai, A. Shehu, and H. Jang. Computational structural biology: The challenges ahead. *Molecules*, 24(3):637, 2018.

[13] B. Olson and A. Shehu. Multi-objective stochastic search for sampling local minima in the protein energy surface. In *ACM Conf on Bioinf and Comp Biol (BCB)*, pages 430–439, Washington, D. C., September 2013.

[14] G. E. Schwartz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

[15] A. Shehu. Probabilistic search and optimization for protein energy landscapes. In S. Aluru and A. Singh, editors, *Handbook of Computational Molecular Biology*. Chapman & Hall/CRC Computer & Information Science Series, 2013.

[16] L. Van Der Maaten, E. Postma, and J. Van den Herik. Dimensionality reduction: a comparative review. *J Mach Learn Res*, 10(66-71):13, 2009.

[17] G. Zhang, L. Ma, X. Wang, and X. Zhou. Secondary structure and contact guided differential evolution for protein structure prediction. *IEEE/ACM Trans Comput Biol and Bioinf*, 2018. preprint.

[18] G. J. Zhang, G. Zhou, X, X. F. Yu, H. Hao, and L. Yu. Enhancing protein conformational space sampling using distance profile-guided differential evolution. *IEEE/ACM Trans Comput Biol and Bioinf*, 14(6):1288–1301, 2017.

[19] J. Zhang and D. Xu. Fast algorithm for clustering a large number of protein structural decoys. In *Intl Conf on Bioinf and Biomed (BIMB)*, pages 30–36. IEEE, 2011.