ProtNN: Fast and Accurate Nearest Neighbor Protein Function Prediction based on Graph Embedding in Structural and Topological Space

Wajdi Dhifli and Abdoulaye Baniré Diallo

Department of Computer Science, University of Quebec At Montreal
PO box 8888, Downtown station, H3C 3P8, Montreal (Quebec) Canada,
dhfli.wajdi@courrier.uqam.ca, diallo.abdoulaye@uqam.ca

Abstract. Studying the function of proteins is important for understanding the molecular mechanisms of life. The number of publicly available protein structures has increasingly become extremely large. Still, the determination of the function of a protein structure remains a difficult, costly, and time consuming task. The difficulties are often due to the essential role of spatial and topological structures in the determination of protein functions in living cells. In this paper, we propose ProtNN, a novel approach for protein function prediction. Given an unannotated protein structure and a set of annotated proteins, ProtNN finds the nearest neighbor annotated structures based on protein graph pairwise similarities. ProtNN assigns to the query protein the function with the highest number of votes across the set of \( k \) nearest neighbor reference proteins, where \( k \) is a user defined parameter. Experimental evaluation demonstrates that ProtNN is able to accurately classify several datasets in an extremely fast runtime compared to state-of-the-art approaches.

Keywords: Protein structures, function prediction, graph classification

1 Introduction

Proteins are ubiquitous in the living cells. They play key roles in the functional and evolutionary machinery of species. Studying protein functions is paramount for understanding the molecular mechanisms of life. The advances in sequencing techniques provide high amounts of biological data including protein structures. In fact, the number of proteins in the Protein Data Bank (PDB) [1] has more than tripled over the last decade. Alternative databases such as SCOP [2] and CATH [3] are undergoing the same trend. However, the determination of the function of protein structures remains a difficult, costly, and time consuming task. Manual protein functional classification methods are no longer able to follow the rapid increase of data. Accurate computational and machine learning tools present an efficient alternative that could offer considerable boosting to meet the increasing load of data.

Proteins are composed of complex three-dimensional folding of long chains of amino acids. This spatial structure is an essential component in protein functionality and is thus subject to evolutionary pressures to optimize the inter-residue
contacts that support it [1]. Existing computational methods for protein function prediction try to simulate biological phenomena that define the function of protein. The most conventional technique is to perform a similarity search between an unknown protein and a reference database of proteins with known functions. The query protein is assigned with the same functional class of the most similar (based on the sequence or the structure) reference protein. There exists several classification methods based on the protein sequence (e.g. Blast [5], ...); or the protein structure (e.g. Combinatorial Extension [6], Sheba [7], FatCat [8], ...). These methods rely on the assumption that protein sharing the most common sites are more likely to share functions. This classification strategy is based on the hypothesis that structurally similar proteins could share a common ancestor [9].

Another popular approach for protein functional classification is to look for relevant substructures (also so-called motifs) among proteins with known functions, then use them as features to identify the function of unknown proteins. Such motifs could be discriminative [10], representative [11], cohesive [4], etc. Each of the mentioned protein functional classification approaches suffers different drawbacks. Sequence-based classification do not incorporate spatial information of amino acids that are not contiguous in the primary structure but interconnected in 3D space. This makes them less efficient in predicting the function for structurally similar proteins with low sequence similarity (remote homologues). Both structure and substructure-based classification techniques do incorporate spatial information in function prediction which makes them more efficient than sequence-based classification. However, such consideration makes these methods subject to the "no free lunch" principle [12], where the gain in accuracy comes with an offset of computational cost. Hence, it is essential to find an efficient way to incorporate 3D structure information with low time complexity.

In this paper, we present ProtNN, a novel approach for function prediction of protein 3D structures. ProtNN incorporates protein 3D structure information via the combination of a rich set of structural and topological descriptors that guarantee an informative multi-view representation of the structures that considers spatial information through different dimensions. Such a representation transforms the complex protein 3D structure into an attribute vector of fixed size allowing computational efficiency. For classification, ProtNN assigns to a query protein the function with the highest number of votes across the set of its $k$ most similar reference proteins, where $k$ is a user defined parameter. Experimental evaluation shows that ProtNN is able to accurately classify different benchmark datasets with a gain of up to 47x of computational cost compared to gold standard approaches from the literature.

2 Methods

2.1 Graph Representation of Protein 3D Structures

A crucial step in computational studies of protein 3D structures is to look for a convenient representation of their spatial conformations. Graphs represent the most appropriate data structure to model the complex structure of protein. In
this context, a protein 3D structure can be seen as a set of elements (amino acids and atoms) that are interconnected through chemical interactions [4,9,11]. Figure 1 shows a real example of the human hemoglobin protein and its graph representation. The figure shows clearly that the graph representation preserves the overall structure of the protein and its components.

Fig. 1. The human hemoglobin protein 3D-structure (PDBID: 1GZX) and its corresponding graph. Nodes and edges represent, respectively, amino acids from the structure and spatial links between them. Blue edges represent the primary structure and gray edges are spatial links between distant amino acids.

Protein Graph Model Let $G$ be a graph consisting of a set of nodes $V$ and edges $E$. $L$ is a label function that associates a label $l$ to each node in $V$. Each node of $G$ represents an amino acid from the 3D structure, and is labeled with its corresponding amino acid type. Let $\Delta$ be a function that computes the euclidean distance between pairs of nodes $\Delta(u, v), \forall u, v \in V$, and $\delta$ a distance threshold. Each node in $V$ is defined by its 3D coordinates in $\mathbb{R}^3$, and both $\Delta$ and $\delta$ are expressed in angstroms ($\text{Å}$). Two nodes $u$ and $v$ ($\forall u, v \in V$) are linked by an edge $e(u, v) \in E$, if the distance between their $C_\alpha$ atoms is below or equal to $\delta$. Formally, the adjacency matrix $A$ of $G$ is defined as follows:

$$
A_{u,v} = \begin{cases} 
1, & \text{if } \Delta(C_{\alpha u}, C_{\alpha v}) \leq \delta \\
0, & \text{otherwise}
\end{cases}
$$

(1)

2.2 Structural and Topological Embedding of Protein Graphs

Graph Embedding In ProtNN, each protein 3D structure is represented by a graph according to Equation 1. Then, each graph is embedded into a vector of structural and topological features under the assumption that structurally similar graphs should give similar structural and topological feature-vectors. In such manner, ProtNN guarantees accuracy and computational efficiency. It is worth noting that even though structurally similar graphs should have similar
Structural and Topological Attributes In ProtNN, the pairwise similarity between two protein graphs is measured by the distance between their vector representations. In order to avoid the loss of structural information in the embedding, and to guarantee ProtNN accuracy, we use a set of structural and topological attributes from the literature that have shown to be interesting and efficient in describing connected graphs [13,14]. In the following, we present the list of the used attributes, see the Appendix for formal definitions: (A1) number of nodes, (A2) number of edges, (A3) average degree, (A4) density, (A5) average clustering coefficient, (A6) average effective eccentricity, (A7) effective diameter, (A8) effective radius, (A9) closeness centrality, (A10) percentage of central nodes, (A11) percentage of end points, (A12) number of distinct eigenvalues, (A13) spectral radius, (A14) second largest eigenvalue, (A15) energy, (A16) neighborhood impurity, (A17) link impurity, and (A18) label entropy.

2.3 ProtNN: Nearest Neighbor Protein Functional Classification

The general classification pipeline of ProtNN can be described as follows: first a preprocessing is performed on the reference protein database $\Omega$ in which a graph model $G_P$ is created for each reference protein $P$, $\forall P \in \Omega$, according to Equation 1. A structural and topological description vector $V_P$ is created for each graph model $G_P$, by computing the corresponding values of each of the structural and topological attributes described in Section 2.2. The resulting matrix $M_\Omega = \bigcup V_P, \forall P \in \Omega$, represents the preprocessed reference database that is used for prediction in ProtNN. In order to guarantee an equal participation of all used attributes in the classification, a min-max normalization ($x_{\text{normalized}} = \frac{x - \text{min}}{\text{max} - \text{min}}$, where $x$ is an attribute value, min and max are the minimum and maximum values for the attribute vector) is applied on each attribute of $M_\Omega$ independently such that no attribute will dominate in the prediction. It is also worth mentioning that for real world applications $M_\Omega$ is only computed once, and can be incrementally updated with other attributes as well as newly added protein 3D structures with no need to recompute the attributes for the entire set. This guarantees a high flexibility and easy extension of ProtNN in real world application. The prediction step in ProtNN is described in Algorithm 1.

3 Experiments

3.1 Datasets

We use six benchmark datasets of protein structures that have previously been used in [10,15,16,17]. Each dataset is composed of positive protein examples.
**Algorithm 1: ProtNN (The prediction step)**

**Data:** $Q$: Query protein 3D structure, $M_{\Omega}$: Description matrix of the reference database of protein 3D structures, $k$: number of similar proteins

**Result:** $C_Q$: Functional class of $Q$

```
begin
1. $G_Q \leftarrow$ create a graph model for $Q$ according to Equation 1;
2. $V_Q \leftarrow G_Q$ is embedded into a vector $V$ using the attributes;
3. $NN_Q^k \leftarrow \emptyset$;
4. $Vdist_Q \leftarrow \emptyset$;
5. foreach ($P$ in $M_{\Omega}$) do
   6. $Vdist_Q[P] \leftarrow$ distance($V_Q$, $P$);  // The distance between vectors of query protein $Q$ and the reference protein $P$.
   7. $NN_Q^k \leftarrow Top_k(Vdist_Q)$;  // Select the $k$ nearest reference protein neighbors
   8. $C_Q \leftarrow$ The functional class with the highest number of votes across the set of $NN_Q^k$ reference proteins;
```

---

Table 1. Characteristics of the experimental datasets. SCOP ID: identifier of protein family in SCOP, Pos.: number of positive examples, Neg.: number of negative examples.

| Dataset | SCOP ID | Family name                | Pos. | Neg. |
|---------|---------|----------------------------|------|------|
| DS1     | 48623   | Vertebrate phospholipase A2| 29   | 29   |
| DS2     | 52592   | G-proteins                 | 33   | 33   |
| DS3     | 48942   | C1 set domains             | 38   | 38   |
| DS4     | 56437   | C-type lectin domains      | 38   | 38   |
| DS5     | 56251   | Proteasome subunits        | 35   | 35   |
| DS6     | 88854   | Protein kinases, catalytic subunits | 41 | 41 |

that are from a selected protein family, and negative protein examples that are randomly sampled from the PDB [1]. The selected positive protein families are Vertebrate phospholipase A2, G-protein family, C1-set domains, C-type lectin domains, and protein kinases, catalytic subunits. Table 1 summarizes the characteristics of the six datasets.

### 3.2 Protocol and Settings

Experiments were performed on CentOS Linux workstations with Intel core i7 CPU at 3.40 GHz, and 16.00 GB of RAM. To transform a protein into graph, we used a $\delta$ value of 7 Å. The evaluation measure is the classification accuracy, and the evaluation technique is Leave-One-Out (LOO) where each dataset is used to create $N$ classification scenarios, where $N$ is the number of proteins in the dataset. In each scenario, a reference protein is used as a query instance and the rest of the dataset is used as reference. The aim is to correctly predict the class of the query protein. The classification accuracy for each dataset is averaged over results for all the $N$ evaluations.
4 Results and Discussion

4.1 ProtNN Classification Results

Results Using Different Distance Measures We study the effect of varying the distance measure on the classification accuracy of ProtNN. We fixed $k=1$, and we used nine different distance measures namely Euclidean, standardized Euclidean (std-euclidean), Cosine, Manhattan, Correlation, Minkowski, Chebyshev, Canberra, and Braycurtis. See [18] for a formal definition of these measures. Figure 2 shows the obtained results. Overall, varying the distance measure did not significantly affect the classification accuracy of ProtNN for the six datasets. Indeed, the standard deviation of the classification accuracy of ProtNN with each distance measure did not exceed 3% on the six datasets. A ranking based on the average classification accuracy over the six datasets suggests the following descending order: (1) Manhattan, (2) Braycurtis, (3) std-Euclidean, (4) Canberra, (5) Cosine, (6) Euclidean - Minkowski, (8) Correlation, (9) Chebyshev.

Fig. 2. Classification accuracy of ProtNN using different distance measures ($k=1$).

Results Using Different Numbers of Nearest Neighbors In the following, we evaluate the classification accuracy of ProtNN on each of the six benchmark datasets using different numbers of nearest neighbors $k \in [1,10]$. The same experiment is performed using each of the top-five distance measures. For simplicity, we plot the average value of classification accuracy for each value of $k \in [1,10]$ over the six datasets using each of the top-five measures. Figure 3 shows the obtained results. The number of considered nearest neighbors $k$ has a clear effect on the accuracy. The obtained results suggest that the "optimal" value of $k \in \{1,2\}$. The overall accuracy tendency shows that it decreases with higher values of $k$. This is due to the structural similarity that a query protein may share with other evolutionary close proteins exerting different functions. High values of $k$ engender considering too many neighbors which may causes a misclassification.

Analysis of the Used Attributes In the following, we study the importance of the used attributes in order to identify which ones are the most informative. We
Fig. 3. Tendency of average accuracy of ProtNN for each value of $k \in [1,10]$ over the six datasets and using each of the top-five distance measures. The dashed line represents the linear tendency of the results.

Table 2. Empirical ranking of the structural and topological attributes.

| Dataset | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 | A9 | A10 | A11 | A12 | A13 | A14 | A15 | A16 | A17 | A18 |
|---------|----|----|----|----|----|----|----|----|----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| DS1     | 0  | 2  | 1  | 6  | 2  | 1  | 5  | 9  | 10 | 2   | 8   | 6   | 13  | 16  | 17  | 12  | 17  |    |
| DS2     | 8  | 12 | 15 | 16 | 18 | 4  | 9  | 16 | 23 | 17  | 9   | 21  | 14  | 25  | 17  | 23  | 9   |    |
| DS3     | 8  | 13 | 2  | 6  | 17 | 10 | 16 | 11 | 14 | 4   | 8   | 18  | 21  | 2  | 21  | 23  | 9   | 18  |
| DS4     | 4  | 7  | 21 | 17 | 20 | 6  | 11 | 17 | 16 | 7   | 2   | 14  | 21  | 22  | 20  | 21  | 24  | 17  |
| DS5     | 12 | 12 | 8  | 10 | 12 | 5  | 7  | 7  | 17 | 17  | 7   | 23  | 23  | 9   | 23  | 9   | 19  | 18  |
| DS6     | 5  | 11 | 9  | 8  | 11 | 6  | 14 | 14 | 13 | 6   | 1   | 17  | 14  | 18  | 24  | 10  | 17  | 13  |
| **Total** | 37 | 57 | 56 | 63 | 80 | 32 | 62 | 74 | 90 | 52  | 29  | 101 | 96  | 78  | 126 | 97  | 104 | 92  |

| **Score** | 0.25 | 0.38 | 0.37 | 0.42 | 0.53 | 0.21 | 0.41 | 0.49 | 0.6 | 0.35 | 0.19 | 0.67 | 0.64 | 0.52 | 0.84 | 0.65 | 0.69 | 0.61 |

| **Rank** | 16 | 13 | 14 | 11 | 8  | 17 | 12 | 10 | 7  | 15  | 18  | 3   | 5   | 9   | 1   | 4   | 2   | 6   |

follow the Recursive Feature Elimination (RFE) approach [19] with ProtNN as the classifier. In RFE, one feature is removed at each iteration, where the remaining features are the ones that best enhance the classification accuracy. The pruning stops when no further enhancement is observed or no more features are left. The remaining features constitute the optimal subset for that context. In Table 2, we record the ranking of the used attributes in our experiments. For more generalization, RFE was performed on each of the six datasets using a combination of each of the top-five distance measures and each of the top-five values of $k$. The total number of experiments is 150. For each attribute, we count the total number of times it appears in the optimal subset of attributes. A score of $\frac{\text{total count}}{\text{number of experiments}}$ is assigned to each attribute according to its total count. It is clear that best subset of attributes depends on the dataset. The five most informative attributes are respectively: A15 (energy), A17 (link impurity), A12 (number of distinct eigenvalues), A16 (neighborhood impurity), and A13 (spectral radius). All spectral attributes showed to be very informative. Indeed, three of them (A15, A12, and A13) ranked in the top-five, and A14 (second largest eigenvalue) ranked in the top-ten (9th) with a score of 0.52 meaning that for more than half of all the experiments, all spectral attributes were selected in
the optimal subset of attributes. Unsurprisingly, A11 (percentage of end points) ranked last with a very low score. This is because proteins are dense molecules and thus very few nodes of their respective graphs will be end points (extremity amino acids in the primary structure with no spatial links). Label attributes also showed to be very informative. Indeed, A17, A16, and A18 ranked respectively 2nd, 4th, and 6th with scores of more than 0.61. This is due to the importance of the distribution of the types of amino acids and their interactions. Both have to follow a certain harmony in order to exert a particular function. A9 (closeness centrality), A5 (average clustering coefficient) and A8 (effective radius) ranked in the top-ten with scores of more than 0.5 (A8 scored 0.49 ≃ 0.5). However, all A1 (number of nodes), A2 (number of edges), A3 (average degree), A4 (density), A6 (average effective eccentricity), A7 (effective diameter), and A10 (percentage of central nodes) scored less than 0.5. This is because each of these attributes is represented by one of the top-ten attributes and thus presents a redundant information. A6 and A9 are both expressed based on all shortest paths of the graph. Both A7 and A8 are expressed based on A6. A10 is expressed based on A8 and thus on A6 too. A1, A2, A3, and A4 are all highly correlated to A5.

Comparison with Other Classification Techniques

We compare our approach with multiple state-of-the-art approaches for protein function prediction namely: sequence alignment-based classification (using Blast [5]), structural alignment-based classification (using Combinatorial Extension (CE) [6], Sheba [7], and FatCat [8]), and substructure(subgraph)-based classification (using GAIA [17], LPGBCMP [15], and D&D [10]). For sequence and structural alignment-based classification, we align each protein against all the rest of the dataset. We assign to the query protein the function of the reference protein with the best hit score. For the substructure-based approaches, all the selected approaches are mainly for mining discriminative subgraphs. LPGBCMP is used with \( max_{var} = 1 \) and \( d = 0.25 \) for, respectively, feature consistency map building and overlapping. For all these approaches, the discovered substructures are considered as features for describing each example of the original data. The constructed description matrix is used for training in the classification. For our approach, we show the classification accuracy results of ProtNN with RFE using std-Euclidean distance. We also show the best results of ProtNN (denoted ProtNN*) with RFE using each of the top-five distance measures. We use \( k = 1 \) both for ProtNN and ProtNN*. Table 3 shows the obtained results.

The alignment-based approaches FatCat and Sheba outperformed CE, Blast, and all the subgraph-based approaches. Except CE, all the other approaches scored on average better than Blast. This shows that the spatial information constitutes an important asset for functional classification. For the subgraph-based approaches, D&D scored better than LPGBCMP and GAIA on all cases except with DS1 where GAIA scored best. On average, ProtNN* ranked first with the smallest distance between its results and the best obtained accuracies with each dataset. This is because ProtNN considers both structural information, and hidden topological properties that are omitted by the other approaches.
Table 3. Accuracy comparison of ProtNN with other classification techniques.

| Dataset | Classification approach | Blast | Sheba | FatCat | CE | LPGBCMP | D&D | GAIA | ProtNN | ProtNN* |
|---------|-------------------------|-------|-------|--------|----|---------|-----|------|--------|---------|
| DS1     | 87.93                   | 81.03 | 100   | 44.82  | 87.50 | 100     | 96.55 | 96.55 |
| DS2     | 81.81                   | 86.36 | 89.39 | 48.48  | 72.50 | 76      | 66   | 80.3  | 89.39  |
| DS3     | 89.94                   | 94.74 | 84.21 | 59.21  | 90.15 | 96      | 89   | 96.05 | 97.37  |
| DS4     | 76.31                   | 92.11 | 100   | 46.05  | 90.00 | 92.5    | 89   | 97.37 | 97.37  |
| DS5     | 85.71                   | 98.57 | 94.28 | 73.71  | 86.80 | 88.5    | 72   | 90    | 94.29  |
| DS6     | 78.04                   | 100   | 93.9  | 80.48  | 90.50 | 95      | 87   | 96.34 | 96.34  |
| Avg. Acc.  | 83.29±5.5  | 92.14±7.3 | 94.65±16.1 | 59.13±15.6 | 86.24±6.9 | 90.17±7.4 | 83.81±12.5 | 92.77±6.7 | 95.22±3.07 |
| Avg. Dist.  | 14.27±7.0  | 5.42±7.2 | 3.93±5.2 | 38.43±15.0 | 11.31±3.3 | 7.39±4.1 | 13.72±9.8 | 4.79±3.2 | 2.34±1.9 |

1Average classification accuracy of each classification approach over the six datasets.
2Average of distances between accuracy of each approach and the best obtained accuracy with each dataset.

4.2 Scalability and Runtime Analysis

In this section, we study the computational cost of ProtNN and FatCat, the most competitive approach. We analyze the variation of runtime for both approaches with higher numbers of protein 3D-structures ranging from 10 to 100 proteins with a step-size of 10. In Figure 4, we report the runtime results in seconds (left) and in log10-scale (right). A huge gap is clearly observed between the runtime of ProtNN and that of FatCat. The gap gets larger with higher numbers of proteins. Indeed, FatCat runtime took over 5570 seconds with the 100 proteins while ProtNN runtime did not exceed 118 seconds for the same set which means that our approach is 47x faster than FatCat on that experiment. The average runtime of graph transformation of ProtNN was 0.8 second and that of the computation of attributes was 0.6 second for each protein. The total runtime of similarity search and function prediction of ProtNN was only 0.1 on the set of 100 proteins. In real world applications, the graph transformation and attribute computation for the reference database are computed only once and can be updated with no need to recompute the existing values. This ensures computational efficiency and easy extension of our approach.

Fig. 4. Runtime comparison of ProtNN and FatCat.
5 Conclusion

In this paper, we proposed ProtNN, a new fast and accurate approach for protein function prediction. We defined a graph transformation and embedding model that incorporates explicit as well as hidden structural and topological properties of the 3D-structure of proteins. We successfully implemented the proposed model and we experimentally showed that it allows to detect similarity and to predict the function of protein 3D-structures efficiently. Empirical results of our experiments showed that considering structural information constitutes a major asset for identifying the functions of proteins correctly. They also showed that the alignment-based classification as well as subgraph-based classification present very competitive approaches. Yet, as the number comparisons between pairs of proteins grows tremendously with the size of dataset, enormous computational costs would be the results of more detailed models. ProtNN showed that it is able to accurately classify multiple benchmark datasets from the literature with very low computational costs.

In future works, we aim to integrating more protein information in our model to further enhance the accuracy of our function prediction system. We also plan to apply our approach on large scale dataset that includes the entire PDB, which will takes ProtNN beyond theoretical proposition to become a reference bioinformatics tool for real world applications.

References

1. Berman, H.M., Westbrook, J.D., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E.: The protein data bank. Nucleic Acids Research 28(1) (2000) 235–242
2. Andreeva, A., Howorth, D., Chandonia, J.M., Brenner, S.E., Hubbard, T.J.P., Chothia, C., Murzin, A.G.: Data growth and its impact on the scop database: new developments. Nucleic Acids Research 36(1) (2008) D419–D425
3. Sillitoe, I., Lewis, T.E., Cuff, A.L., Das, S., Ashford, P., Dawson, N.L., Furnham, N., Laskowski, R.A., Lee, D., Lees, J.G., Lehtinen, S., Studer, R.A., Thornton, J.M., Orengo, C.A.: CATH: comprehensive structural and functional annotations for genome sequences. Nucleic Acids Research 43(Database-Issue) (2015) 376–381
4. Meysman, P., Zhou, C., Cule, B., Goethals, B., Laukens, K.: Mining the entire protein databank for frequent spatially cohesive amino acid patterns. BioData Mining 8 (2015) 4
5. Altschul, S., Gish, W., Miller, W., Myers, E., Lipman, D.: Basic local alignment search tool. Journal of Molecular Biology 215 (1990) 403–410
6. Shindyalov, I.N., Bourne, P.E.: Protein structure alignment by incremental combinatorial extension of the optimum path. Protein Engineering 11(9) (1998) 739–747
7. Jung, J., Lee, B.: Protein structure alignment using environmental profiles. Protein Engineering 13 (2000) 535–543
8. Ye, Y., Godzik, A.: Flexible structure alignment by chaining aligned fragment pairs allowing twists. Bioinformatics 19 (2003) 246–255
9. Borgwardt, K.M., Ong, C.S., Schönauer, S., Vishwanathan, S.V.N., Smola, A.J., Kriegel, H.: Protein function prediction via graph kernels. In: Proceedings Thirteenth International Conference on Intelligent Systems for Molecular Biology 2005, Detroit, MI, USA, 25-29 June 2005. (2005) 47–56
10. Zhu, Y., Yu, J.X., Cheng, H., Qin, L.: Graph classification: a diversified discriminative feature selection approach. In: 21st ACM International Conference on Information and Knowledge Management, ACM (2012) 205–214
11. Dhifli, W., Saidi, R., Mephu Nguifo, E.: Smoothing 3D protein structure motifs through graph mining and amino-acids similarities. Journal of Computational Biology 21(2) (2014) 162–172
12. Wolpert, D., Macready, W.G.: No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation 1(1) (1997) 67–82
13. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graphs over time: densification laws, shrinking diameters and possible explanations. In: eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, ACM (2005) 177–187
14. Li, G., Semerci, M., Yener, B., Zaki, M.J.: Effective graph classification based on topological and label attributes. Statistical Analysis and Data Mining 5(4) (2012) 265–283
15. Fei, H., Huan, J.: Boosting with structure information in the functional space: an application to graph classification. In: ACM knowledge discovery and data mining conference (KDD). (2010) 643–652
16. Jin, N., Young, C., Wang, W.: Graph classification based on pattern co-occurrence. In: ACM International Conference on Information and Knowledge Management. (2009) 573–582
17. Jin, N., Young, C., Wang, W.: GAIA: graph classification using evolutionary computation. In: Proceedings of the 2010 ACM SIGMOD International Conference on Management of data, ACM (2010) 879–890
18. Sergio J. Rojas G., Erik A Christensen, F.J.B.S.: Learning SciPy for Numerical and Scientific Computing - Second Edition. Community experience distilled. Packt Publishing (2015)
19. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine Learning 46(1-3) (2002) 389–422

Appendix: Structural and Topological Attributes

In the following is the list of structural and topological attributes used in PROTNN:

A1- **Number of nodes**: The total number of nodes of the graph $|V|$.
A2- **Number of edges**: The total number of edges of the graph $|E|$.
A3- **Average degree**: The degree of a node $u$, denoted $\text{deg}(u)$, is the number of nodes adjacent to $u$. The average degree of a graph $G$ is the average all $\text{deg}(u)$, $\forall u \in G$. Formally: $\text{deg}(G) = \frac{1}{|V|} \sum_{i=1}^{|V|} \text{deg}(u_i)$.
A4- **Density**: The density of a graph $G = (V, E)$ measures how many edges are in $E$ compared to the maximum possible number of edges between the nodes in $V$. Formally: $\text{den}(G) = \frac{|E|}{(|V| \cdot (|V| - 1)/2)}$.
A5- **Average clustering coefficient**: The clustering coefficient of a node $u$, denoted $c(u)$, measures how complete the neighborhood of $u$ is, $c(u) = \frac{k_u}{e_u}$, where $k_u$ is the number of neighbors of $u$ and $e_u$ is the number of connected pairs of neighbors. The average clustering coefficient of a graph $G$, is given as the average value over all its nodes. Formally: $\overline{C(G)} = \frac{1}{|V|} \sum_{i=1}^{|V|} c(u_i)$. 
A6- **Average effective eccentricity**: For a node $u$, the effective eccentricity represents the maximum length of the shortest paths between $u$ and every other node $v$ in $G$, $e(u) = \max\{d(u, v) : v \in V, u \neq v\}$. The average effective eccentricity is defined as $\overline{e}(G) = \frac{1}{|V|} \sum_{u=1}^{|V|} e(u)$.

A7- **Effective diameter**: It represents the maximum value of effective eccentricity over all nodes in the graph $G$, i.e., $\text{diam}(G) = \max\{e(u) | u \in V\}$ where $e(u)$ represents the effective eccentricity of $u$ as defined above.

A8- **Effective radius**: The effective radius represents the minimum value of effective eccentricity over all nodes of $G$, $\text{rad}(G) = \min\{e(u) | u \in V\}$.

A9- **Closeness centrality**: The closeness centrality measures how fast information spreads from a given node to other reachable nodes in the graph. For a node $u$, it represents the reciprocal of the average shortest path length between $u$ and every other reachable node in the graph $G$, $C_c(u) = \frac{|V| - 1}{\sum_{v \in (V \setminus u)} d(u, v)}$ where $d(u, v)$ is the length of the shortest path between the nodes $u$ and $v$. For $G$, we consider the average value of closeness centrality of all its nodes, $C_c(G) = \frac{1}{|V|} \sum_{u=1}^{|V|} C_c(u)$.

A10- **Percentage of central nodes**: It is the ratio of the number of central nodes from the number of nodes in the graph. A node $u$ is central if the value of its eccentricity is equal to the effective radius of the graph, $e(u) = \text{rad}(G)$.

A11- **Percentage of end points**: It represents the ratio of the number of nodes with $\text{deg}(u) = 1$ from the total number of nodes of $G$.

A12- **Number of distinct eigenvalues**: The adjacency matrix $A$ of $G$ has a set of eigenvalues. We count the number of distinct eigenvalues of $A$.

A13- **Spectral radius**: Let $\lambda_1, \lambda_2, ..., \lambda_m$ be the set of eigenvalues of the adjacency matrix $A$ of $G$. The spectral radius of $G$, denoted $\rho(G)$, represents the largest magnitude eigenvalue, $\rho(G) = \max(\{ |\lambda_i| \} | i \in \{1, ..., m\}$.

A14- **Second largest eigenvalue**: The value of the second largest eigenvalue.

A15- **Energy**: The energy of an adjacency matrix $A$ of a graph $G$ is defined as the squared sum of the eigenvalues of $A$. Formally: $E(G) = \sum_{i=1}^{m} \lambda_i^2$.

A16- **Neighborhood impurity**: For a node $u$ having a label $L(u)$ and a neighborhood $N(u)$, it is defined as $\text{ImpDeg}(u) = |L(v) : v \in N(u), L(u) \neq L(v) |$. The neighborhood impurity of $G$ is the average $\text{ImpDeg}$ over all nodes.

A17- **Link impurity**: An edge $\{u, v\}$ is considered to be impure if $L(u) \neq L(v)$. The link impurity of a graph $G$ with $k$ edges is defined as: \(|\{u,v\} \in E, L(u) \neq L(v) | / k\).

A18- **Label entropy**: It measures the uncertainty of labels. For a graph $G$ of $k$ labels, it is defined as $E(G) = -\sum_{i=1}^{k} p(l_i) \log p(l_i)$, where $l_i$ is the $i^{th}$ label.