Predicting Protein Structural Features with NeuroEvolution of Augmenting Topologies

Bruno Grisci
Institute of Informatics
Federal University of Rio Grande do Sul
Porto Alegre, RS, Brazil
bigrisci@inf.ufrgs.br

Márcio Dorn
Institute of Informatics
Federal University of Rio Grande do Sul
Porto Alegre, RS, Brazil
mdorn@inf.ufrgs.br

***Abstract*—The study of proteins and the prediction of their three-dimensional structure is one of the most challenging problem in Structural Bioinformatics. Over the last years, several computational strategies have been proposed as a solution to this problem. As revealed by recent CASP experiments, the best results have been achieved by knowledge-based methods. Despite the advances in the development of computational methods, systems and algorithms for solving this complex problem, further research remains to be done. In this paper, we present a new method based on evolving artificial neural networks to extract structural features from experimentally-determined proteins. Structural models can be used to predict the three-dimensional structure of unknown protein sequences and help to reduce the conformational search space of protein molecules. Neural networks using genetic algorithms has shown great promise in this complex learning task. The proposed method has been tested with five protein sequences. The results show that predicted 3-D structures are topologically comparable to their correspondent experimental ones, thus corroborating the effectiveness of our proposal.**

I. Introduction

The understanding of biological processes has been an ongoing challenge for humanity. Currently, one of the main research problems in Structural Bioinformatics is the prediction of the three-dimensional (3-D) structure of proteins. A protein molecule is composed of an ordered linear chain of amino acid residues that performs a variety of functions by assuming a particular three-dimensional shape [20].

Current experimental methods such as NMR and X-ray diffraction for structure determination are not able to cope with the present and future need for protein structure determination. The prediction of the 3-D structure of a protein is experimentally expensive, time-consuming and remains a difficult problem in Structural Bioinformatics [8]. Over the last years, several computational strategies have been proposed as a solution to the *Protein Structure Prediction* (PSP) problem. PSP methods can be organized into four classes [9]: (*a*) first principle methods without database information [23]; (*b*) first principle methods with database information [25]; (*c*) fold recognition methods [6]; and (*d*) comparative modelling methods [21]. Methods *b,c* and *d* are Knowledge-based methods and make use of structural information from X-ray diffraction patterns of crystallized proteins.

However, despite advances, this problem remains computationally expensive due to a large amount of resources needed for the simulation of molecular interactions. As revealed by last CASP[1] experiments, the best results were achieved by knowledge-based methods. These methods are dependent on experimental data and present two main challenges: (1) the development of a computational strategy to identify and retrieve structural information from the *Protein Data Bank* (PDB); and (2) the definition of a knowledge-based search strategy to find native-like protein structures.

In this paper, we propose a new technique to extract structural features from experimentally-determined proteins. Evolving artificial neural networks are applied to find common characteristics in experimental protein databases. These structural models can be used to predict the three-dimensional structure of unknown protein sequences in knowledge-based prediction methods and help to reduce the conformational search space of protein molecules. Neural networks using genetic algorithms has shown great promise in this complex reinforcement learning tasks. Section II presents fundamental concepts of protein structure and its representation and related works. Section III describes the proposed method to acquire structural information from the PDB. In section IV we present the computational experiments and discussion of results. The last section concludes the paper and points out some future works.

II. Preliminaries

A. Protein, structure and its representation

Proteins, also called polypeptides, are polymers formed from 20 different possible amino acid residues. Each protein is defined by a unique sequence of amino acids that under physiological conditions fold into a precise shape known as its native state [4]. The sequence of amino acids is the protein primary structure. Each amino acid residue has an alpha carbon with bonds to amino and carboxyl and a variable side chain that determines the particular physicochemical properties of the amino acid residue. A peptide is a molecule composed of two or more amino acid residues chained by a chemical bond called the peptide bond. This bond is formed when the carboxyl group of one residue reacts with the amino group of the other residue, releasing a water molecule [20]. The interaction between the amino acids in a protein causes the

[1] http://predictioncenter.org

978-1-5090-0620-5/16/$31.00

polypeptide chain to fold, usually in a proper configuration, as a helix or a sheet. This folding pattern is the protein secondary structure [3].

There is two main computational representation of polypeptide tertiary structures, that is, the protein's geometric shape. The first way is to represent the 3-D protein structure with the Cartesian position of the atoms. This model has $3n$ degrees of freedom, n being the number of atoms of the polypeptide structure, and increases if the solvent is considered, so it is computationally expensive. The second representation characterizes the polypeptide structure by the set of dihedral torsion angles and the fact that bond lengths are approximately constant in a polypeptide chain (Fig. 1). That is the representation chosen for our method because it has far fewer degrees of freedom for the representation of the backbone of a polypeptide: $3m$, m being the number of amino acids in the structure.

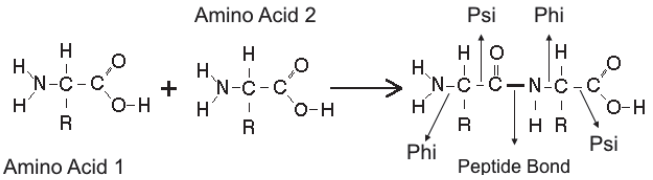


Fig. 1: Peptide Bond. A polypeptide chain can be represented by its set of main-chain and side-chain torsion angles: ϕ (phi), ψ (psi) and χ (chi). The number of χ angles depends on the type of amino acid residue and are represented in the figure by the group R.

In this representation, the 3-D protein structure is encoded by the set of the three dihedral torsion angles of each of its amino acids, since with then it is possible to compute the relative Cartesian position the atoms will need to assume. These angles are called omega, phi, and psi. The peptide bond tends to be planar, with two allowed states for the ω (omega) torsion angle: trans 180.0° (usually), and cis 0° (rarely). Only the ϕ (phi) and ψ (psi) torsion angles need to be used to represent the protein backbone.

B. Related work

A previous attempt to extract protein structural features creating angles intervals from data from protein databases has made with the MOIRAE method [10], with the use of a different approach to extracting information from the protein database, clustering algorithm and neural network training. While the MOIRAE uses k-means for clustering and MLP (Multilayer Perceptron) for training a neural network, our new method uses hierarchical clustering and NEAT [27] to autonomously train different neural networks better suited for the various data retrieved from the database.

The capacity of NEAT finding optimal network topologies has already been shown to return best results than conventional MLP for the feature selection problem [26]. Neural networks, more specifically Deep Convolutional Neural Fields, have also been used for the prediction of the secondary structure of proteins with good results [28].

III. The Proposed Method

Our method is knowledge based and uses information from the Protein Data Bank to create search intervals for the conformational angles phi and psi. The goal is to find intervals inside the range of -180° and 180° for each tuple phi and psi of each amino acid of the target amino acid sequence, so that the interval should contain the real value of the angle, thus restraining the search space. These intervals can be used by different optimization techniques such as Genetic Algorithms or Simulated Annealing to create more efficient, less time-consuming searches since the search space was greatly reduced.

Figure 2 summarizes the proposed method. It can be seen as a sequence of four steps: 1) Database search, 2) Data clustering, 3) Neural Network training, and 4) Intervals creation. The method input is a target amino acid sequence for which we want to find the pairs phi and psi for each amino acid, and its secondary structure sequence, which can be obtained with accuracy from existing methods such as the one used by Stride [12].

The main assumption of our method is that we should be able to learn the expected values for the angles phi and psi for an amino acid in the target amino acid sequence if we use the already existing phi and psi information from other proteins in the PDB, looking not only for occurrences of the amino acid itself but also considering its predecessor and successor, as well as its secondary structure.

We start dividing the target amino acid sequence into segments of size 3 (Tab. I) and looking for matches in the PDB. With the list of proteins that contain the corresponding segment, we download its PDB file and pass it to Stride to get its secondary structure information and phi and psi values. Then, each segment will have a list of points phi and psi from the different proteins in which the segment appears. Thus, we run a clustering algorithm to create different clusters of points phi and psi for each segment (Fig. 3). With each point assigned to a cluster, we create a dataset of the three amino acids of the segment, the three secondary structures of these amino acids and the cluster it belongs. This is used as input for our neural network training; that uses NEAT, a new method that uses genetic algorithms to evolve neural networks (Fig. 5). The neural networks are trained to learn how to classify the amino acids and secondary structure of one of the segments in the clusters found previously. Once the training is complete, the neural network can classify new inputs with generalization. In the final step, we pass the original target amino acids sequence and its secondary structures to the corresponding neural networks, that output a cluster for each one of them. We then create the intervals centering them in mean cluster value and adding and subtracting one and a half standard deviation from it. The final method output is a set of intervals, one for the angles phi of the amino acids in the target sequence, and other for the angles psi.

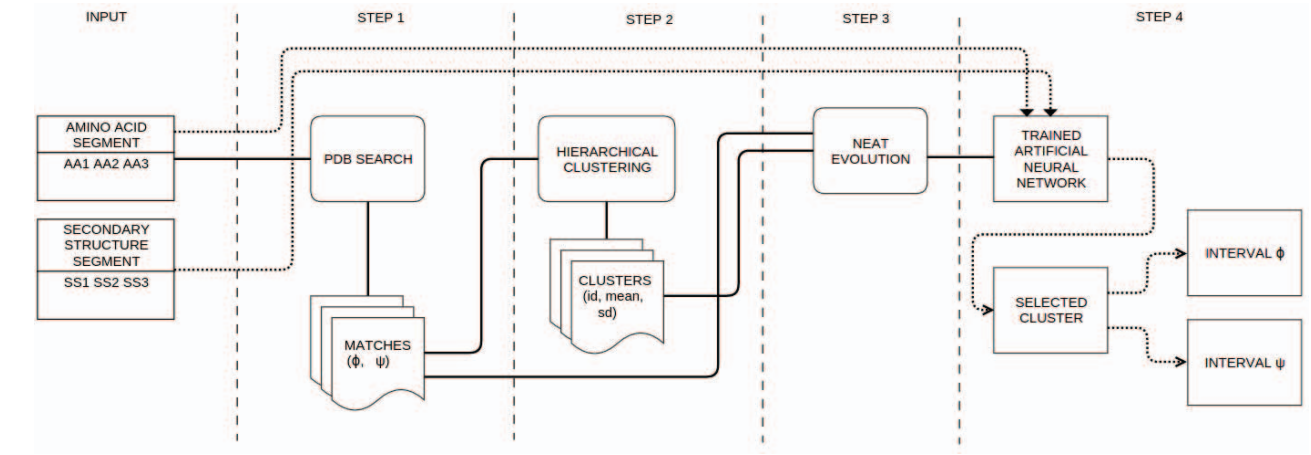


Fig. 2: Method overview for a segment of 3 amino acids. In step 1 we look in the PDB for proteins that contain the segment in its chain, and save them as "matches". We then analyze these matches to extract the corresponding secondary structure and the angles ϕ and ψ of the central amino acid of the segment. In step 2, these pairs of angles are clustered with hierarchical clustering, so each match belongs to a cluster. In step 3 we combine the information of the amino acid segment, secondary structure segment and cluster of each match into a training set and evolve neural networks using NEAT to be able to predict a cluster from the segment of amino acids and secondary structure. In step 4 the final neural network receives as input the original amino acid and secondary structure segments and output a cluster for it. The ϕ and ψ intervals for the segment are then created by adding and subtracting one and a half standard deviations to the ϕ and ψ means from the cluster.

A. Database Search

To find the needed information and limit the range of the final intervals, we divide our target amino acid sequence into segments of size 3 and search the PDB for occurrences of them. So, for a sequence of size n, we will have $n-2$ segments of size 3. Table I shows the segmentation of an amino acid sequence. Each segment will late provide information for its middle amino acid, so we are considering not only the amino acid itself but also its neighbors as a form to learn the expected angles values from the database. The first and last amino acids of the target sequence are not represented by the edges of an amino acid sequence are often unstable.

TABLE I: Construction of amino acids segments of size 3 from a target protein of size n. Each segment represents its central amino acid, written in bold in this table. The first and last amino acids from the target protein are not considered by the method.

Segment	**AA**$_1$	**AA**$_2$	**AA**$_3$	**AA**$_4$	**AA**$_5$	**...**	**AA**$_n$
S_1	AA_1	**AA**$_2$	AA_3	-	-	...	-
S_2	-	AA_2	**AA**$_3$	AA_4	-	...	-
S_3	-	-	AA_3	**AA**$_4$	AA_5	...	-
...							
S_{n-2}	-	-	-	-	-	...	AA_n

For each segment we run the Protein BLAST [2], a program that searches a protein database, in our case the PDB, looking for the occurrences of the segment in the proteins of the database. It returns a list of every protein that has a match with the input segment, so we can download its PDB file and analyze it with STRIDE [12], obtaining its corresponding secondary structure and phi and psi angles for every amino acid. All segments of the original target sequence are matched to lists formed of equal segments from the proteins of the PDB, and this information is saved.

B. Data Clustering

From the database search, we now have, for each of our segments of 3 amino acids from the target amino acid sequence, a list of all the matches from proteins from the PDB, together with its corresponding secondary structures and the phi and psi values for the central amino acid of the segments. The next step is to organize this data by the angle values in the form of clusters, for us to be able to generalize this information later on.

The tuples phi and psi are treated as two-dimensional points with a range of values from -180° to 180° in both axes, with phi being the X axis and psi the Y axis. Due to the cyclical nature of the angles, we need to consider the extremities values as being the same, i.e., the values -180° and 180° are two representations for the same angle value. From now on we should consider it for every calculation, including the distance between points.

1) Hierarchical Clustering: Since the database search returns an unknown number of points distributed in a new fashion for each amino acid, we are not able to predict the number of clusters that best fit the data. For this reason, a method called hierarchical clustering [16] was chosen. This is an iterative process that starts considering each point of the dataset as being a cluster and then proceeds combining the nearer clusters until there is only one cluster containing every point. Using a distance threshold, it is possible to set a stop condition for the method, and it will stop when the distance between the clusters is larger than the threshold, returning the

current clusters. For our method the chosen distance threshold was 90°, since this usually comprehend the distance between different structure regions [13] and generated the best general results. The metric distance used was the Euclidian distance, modified to take into account the cyclical property already discussed. Hierarchical Clustering is then able to classify our data points into an originally unknown number of clusters as desired.

Algorithm 1 Pseudocode for hierarchical clustering

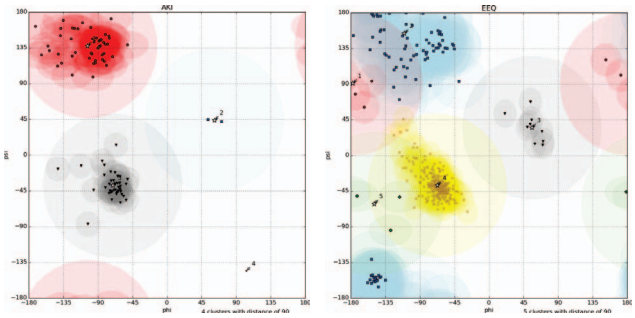
Require: Matrix D of pairwise distances between points, points P, threshold distance t
  Build graph G assingning one vertex to each cluster
  Form $\|P\|$ clusters with one element each
  while there is more than one cluster **and** distance between clusters $> t$ **do**
    Get the two closest clusters C_A and C_B
    Merge C_A and C_B into new cluster C with $\|C_A\|$ + $\|C_B\|$ elements
    Calculate the distance between C and all other clusters

    if the clusters are close **then**
      Add new vertex C to G and connect it to vertices C_A and C_B
      Remove the rows and columns of D corresponding to C_A and C_B
      Add a new row and column to D corresponding to cluster C
    end if
  end while
  return G

We run the clustering algorithm for all segments of the target amino acid sequence, and for all points, it is assigned an integer value that corresponds to a cluster. Clusters with a single point are considered outliers and removed from the next steps of our method. Figure 3 is a graphical representation of two set of clusters for two different amino acids segments. The cyclical property of the clusters becomes evident when checking clusters near the edges, as it is visible that they continue the opposed side of the chart.

C. Neural Networks Training

Once the segments from the database are classified into clusters, it is time to train neural networks to learn how to classify new inputs from which only the three amino acids and its secondary structures are knows, as in the segments from the original target amino acid sequence. To do this, we need a different neural network for each of these segments, since they all have a different set of points and clusters to be analyzed. A traditional method to perform this task would be to use Artifical Neural Networks (ANNs) with a learning method such as backpropagation. But this approach trains ANNs to learn only the weights of the links between the neurons, while the overall topology of the network, i.e., the number of neurons, links and which neuron is connected to another,



(a) AKI (b) EEQ

Fig. 3: Examples of clustering for two amino acids segments of size three from our method, using hierarchical clustering. Each point is a pair ϕ, ψ from a match returned from the PDB search in step 1. Each cluster is here represented by a different color, the lighter colored shadow around the points representing the cluster standard deviation.

remains fixed, and it is the researcher's job to find what is the best topology for each problem. But how to choose only one topology for the issue of classification of new segments of amino acids and secondary structures if each one of them has a unique, particular training set? To address this issue, we selected a method of evolving and teaching ANNs called NeuroEvolution of Augmenting Topologies (NEAT).

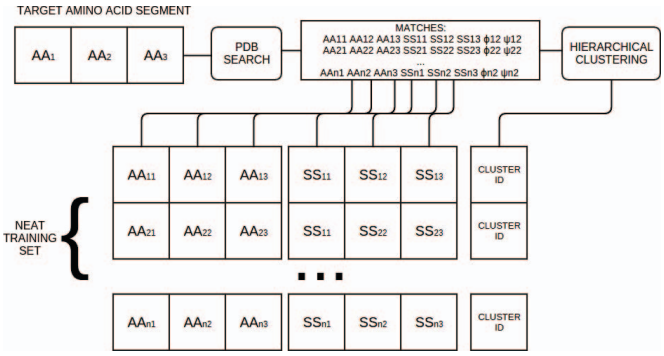


Fig. 4: Diagram of the creation of the training set for the evaluation of the neural networks.

1) NeuroEvolution of Augmenting Topologies: NEAT is a new method that uses Genetic Algorithm to evolve the topology of ANNs. It starts with a random population in which each individual is an ANN and all of them share the same basic topology, i.e., input neurons, output neurons and a link between the input neurons and output neurons with random weights. This minimalistic start condition is important because it assures that unuseful complexity won't be added to the ANNs, since only improvements in the topology of the network that generated better results will be kept. If the method was initialized with random topologies, unuseful neurons or links might be present from the start, and it wouldn't be possible to remove them, what may have a negative effect on the evolution. This also results in smaller, simpler final ANNs.

From this initial population, new populations are generated in an iterative fashion with traditional Genetic Algorithm operators, namely the crossover operation, that combines two individuals from the current population in order to generate a mixed new individual, and mutation, that can change the values of the weights of the links of an ANN, or add new hidden neurons or a new link between current neurons.

A problem that arises from this method is that combining two ANNs with the crossover operation may result in a defective ANN since their topologies may not be compatible with a direct exchange of neurons and links. That is why NEAT uses a historical marking, a numerical value that is assigned to each new piece of structure that appears through the structural mutations. The value of the historical marking is the order in which the structural transformation first appeared along the evolutionary process, and is transferred without changes during the crossover. It allows the NEAT method to match perfectly the same pieces of the topology of two different ANNs, resulting in a functional ANN that respects the organization of its predecessors.

Finally, NEAT uses speciation, also known as a niche, so individuals compete within similar groups of ANNs instead of the totality of the population. This is useful because adding new structure to an ANN usually is disadvantageous without further adjusts, so the speciation gives time to useful topological innovations to evolve, not just to discard them when they first show up. The historical markings are used to determine to which niche an individual belongs.

2) Application: For our method, a target amino acid sequence of length n originates $n-2$ segments of size 3, each one of them corresponding to the middle amino acid and with their set of clusters from the last step. So what we need is to evolve $n-2$ artificial neural networks that receive as input the three amino acids, the three secondary structures and also an extra input called bias that is always set to 1.0, and one output that corresponds to one of the existing clusters. The minimal topology for an ANN, in this case, is, then, seven input neurons linked to one output neuron. Since NEAT uses Genetic Algorithms to evolve the ANNs, it needs a fitness function that evaluates each individual and ranks them by this value. For our method, this fitness is simply a mean squared error: each individual receives as input the information saved in the dataset recovered from the PDB and the output of the ANN is compared to the known cluster value. Figure 4 illustrates the creation of this training set. In the end, the individual with lowest squared error, i.e., the ANN that classified the largest quantity of inputs correctly, is selected. For each segment of the target amino acid sequence, a complete NEAT cycle is performed, using their training set, resulting in $n-2$ different and independent ANNs. This solves the original problem of guessing the best topology for each of the segments since with NEAT it is possible to find different structures and link weights without human intervention. Figure 5 has four full evolved neural networks for four different segments of amino acids. Their final topologies are indeed very different, corroborating the idea that training ANNs with the same topology would not generate the best results.

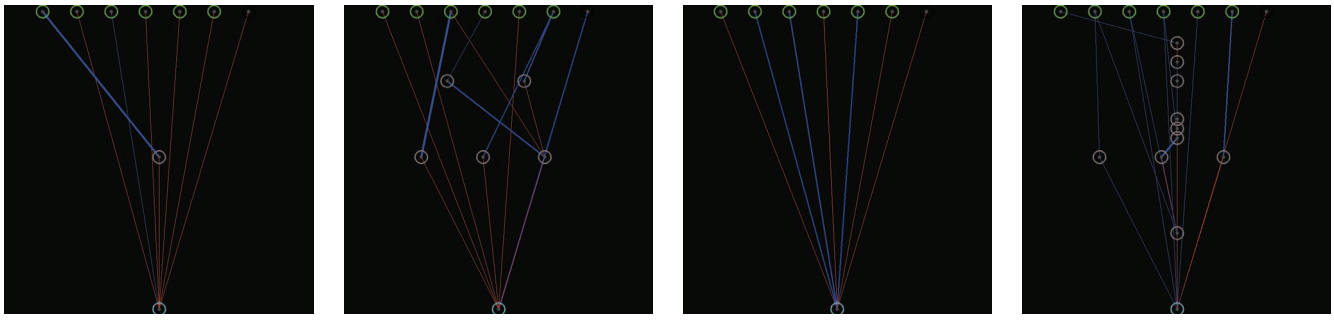
It is important to notice that ANNs work only with numerical inputs and outputs while our method has symbolical inputs and outputs (amino acids, secondary structures, and clusters). The clusters are already defined as integer values, so they don't present a representation problem, but the way one converts the amino acids and secondary structures to numbers may affect the outcome of the learning phase of the ANN, since if numerical values are arbitrarily assigned to each possible input the algorithm may infer correlations and sortings that are not present in nature. To try to avoid this problem we looked for real world properties of amino acids and their secondary structures to make sense of a numerical conversion for the ANNs. For amino acids we used the hydrophobic parameters pi of amino acid side-chains from the partitioning of N-acetyl-amino-acid amides [11], that tend to keep close the amino acids that share the same natural groups, and for secondary structures we assigned values that rank then by their structural similarity and complexity. Hydrophobic residues appearing periodically have already been used for predicting protein secondary structure since hydrophobic affects the stability of secondary structure [14].

D. Intervals Creation

The last step of our method is to produce the search intervals for the angles phi and psi of each amino acid from the original target amino acid sequence. For this, we only need to provide as input to our trained neural networks the amino acids and secondary structures from each of the segments of size 3. Each segment has a matching neural network from the last step, and they will output a cluster. To create the interval for the angle phi of one particular amino acid we extract the mean value of the phi coordinate from the selected cluster. This value is the center of the interval. The boundaries are just the mean added and subtracted by one and a half standard deviations, respecting the cyclicity of the angles. The process to create the interval for angle psi is the same, but using the psi coordinate from the cluster. Repeating this for each one of the segments of the target amino acid sequence of length n will produce $n-2$ phi intervals and $n-2$ psi intervals, corresponding to all amino acids from the sequence except the first and last ones. This is not a problem since the boundaries amino acids position is unstable. For them is assigned intervals with a range from -180° to 180°. Figure 6 show graphical examples of intervals created with our method along the experiments.

IV. Experiments

As we stated in the previous sections, the primary goal of our method is to reduce the protein conformation search space using structural information from experimental-determined proteins. The proposed method (Sec. III) was tested with the amino acid sequence of five proteins obtained from the PDB (Tab. II). These targets were selected to test the proposed method with different sizes and topologies. The target proteins were removed from the training set not to create biased results. For NEAT, we used populations of 300 individuals



(a) AKI (b) EEQ (c) EAL (d) RFY

Fig. 5: Examples of four evolved neural networks using NEAT for four different amino acids segments of size 3. The green nodes are the inputs (amino acids and secondary structures), the black node the bias, the white node the output (cluster) and the gray nodes the hidden layers. The thickness of the lines is proportional to the absolute value of the weight of a link. Blue lines indicate connections with negative weight and red lines positive weights.

and evolved the neural networks for 100 generations. All target protein sequences from Table II were submitted to the proposed method. Torsion angle intervals were built for each amino acid residues. Figure 6 show the predicted intervals. We analyze the quality of computed intervals regarding size (small intervals represents small conformational search space) and if their enclosures the correct torsion angle (experimental protein structure). As could be observed in more than ≈85% os the experiments the computed intervals includes the torsion angle values of the native protein structure. Another important factor to illustrate the quality of calculated intervals is the size o them, the smaller an interval of an amino acid residue, smaller it conformational search space will be.

TABLE II: Amino acids sequences used to test the proposed method. The second column shows the number of residues and the third shows the topology of the target composed by secondary structures components.*Reference not available.

Target Protein	Sequence Length	Secondary Structure Content
1L2Y [22]	20	Two helices
1K43 [24]	14	One sheet
1ROP [5]	63	Two helices
1ZDC [1]	35	Two helices
2PMR*	87	Three helices

We used the Python packages SciPy [17] for the hierarchical clustering and MultiNEAT [7] for NEAT. The experiments were executed in a 64-bit Linux system with 16 GB of RAM memory and a 2.80 GHz CPU with 8 cores. The mean time for clustering a dataset was 4.79 seconds and for evolving a neural network was 68.15 seconds.

V. Analysis and Results

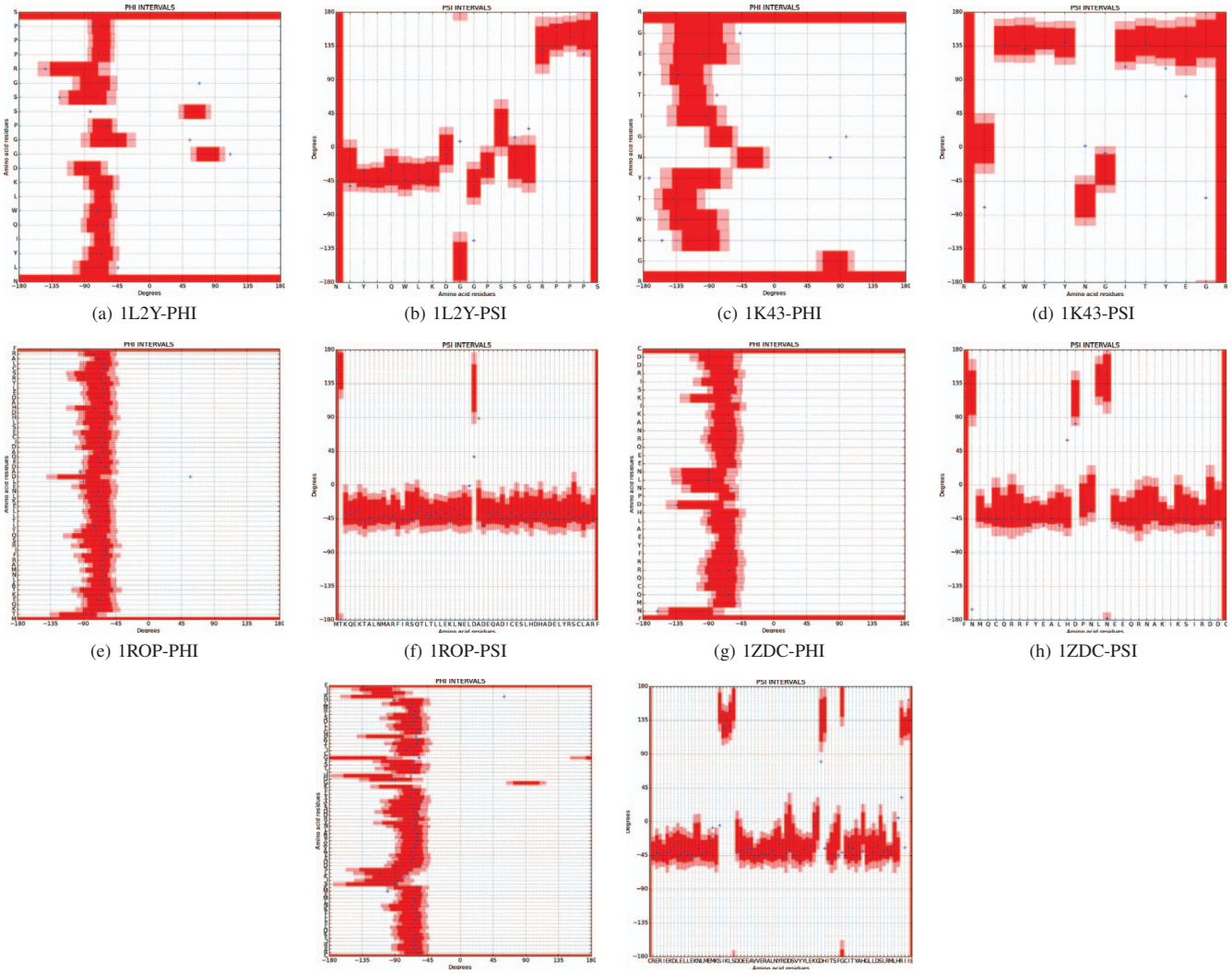
Computed intervals were used to predicted the 3-D structure of the target protein sequences. Figure 7 show the cartoon representation of the experimental and predicted structures. Visual inspection reveals that predicted structures (blue) present a fold comparable with the native three-dimensional structure (red). Torsion angles were chosen without any criteria (center of the interval) with the purpose to demonstrate that predicted intervals include the protein native-like structure. Using intervals we can represent in a more accurate manner the structural information of the experimentally determined protein structures and use this information to guide prediction search methods reducing the conformational search space in the sense that possible torsion angle values of each amino acid residue of a target sequence are limited by the intervals. Coils are irregular and have a more flexible conformation, making the predicition harder. They also affect the formation of sheets, what explain results like the interval prediction for 1K43 (Figure 6c).

TABLE III: Analysis of the intervals from Figure 6. Mean sizes ϕ and ψ are the mean values in degrees of the size of the intervals. The smaller this value, the better, since it reduces the search space of the angles. Enclosure ϕ and ψ are the percentage of torsion angles values of the amino acid residue in the native state that are inside the predicted interval. Higher percentages mean more reliable and accurate predictions.

Protein	Mean size ϕ	Mean size ψ	Enclosure ϕ	Enclosure ψ
1L2Y	51.36°	57.70°	72.22%	83.33%
1K43	75.59°	63.44°	50.00%	58.33%
1ROP	51.97°	55.97°	98.36%	95.08%
1ZDC	58.51°	61.98°	100.0%	93.93%
2PMR	57.39°	55.38°	94.11%	89.41%

Structural quality of predicted structure have been done in terms of the root mean square deviation (RMSD) (Table IV, columns 7) and secondary structure content (Table IV, columns 2-5). The RMSD measure between the C_α of the predicted and experimental structures was calculated using PYMOL routines (Eq. 1). Secondary structure content was computed using PROMOTIF [15], [19]. The secondary structures are also well formed and are close to the experimental 3D structures. RMSD analysis reveals that the proposed method can, without great computational effort, extract reliable structural information from the PDB to be used in knowledge-based prediction methods.

The quality of the evolved neural networks was measured



(a) 1L2Y-PHI (b) 1L2Y-PSI (c) 1K43-PHI (d) 1K43-PSI

(e) 1ROP-PHI (f) 1ROP-PSI (g) 1ZDC-PHI (h) 1ZDC-PSI

(i) 2PMR-PHI (j) 2PMR-PSI

Fig. 6: Torsion angles intervals. Filled boxes represent the predicted torsion angle interval. Blue + characters identify the torsion angle value of the amino acid residue in the native state.

by the success rate in classifying the inputs of the training set in the correct clusters. The average success rate was 79% by neural network with standard deviation of 30%. Another desired feature is that the neural networks aren't too complex, so that they are more efficient and simplier to understand. For this matter NEAT capacity of evolving neural networks from minimal topologies generates really good results. The average number of hidden neurons by neural network, besides the input and output neurons, was 1.78, with standard deviation of 1.12.

$$\mathbf{RMSD}(a,b) = \sqrt{\left(\sum_{i=1}^{n} \|r_{ai} - r_{bi}\|^2\right)/n}, \qquad (1)$$

were r_{ai} and r_{bi} are vectors representing the positions of the same atom i in each of two structures, a and b respectively, and where the structures a and b are optimally superimposed.

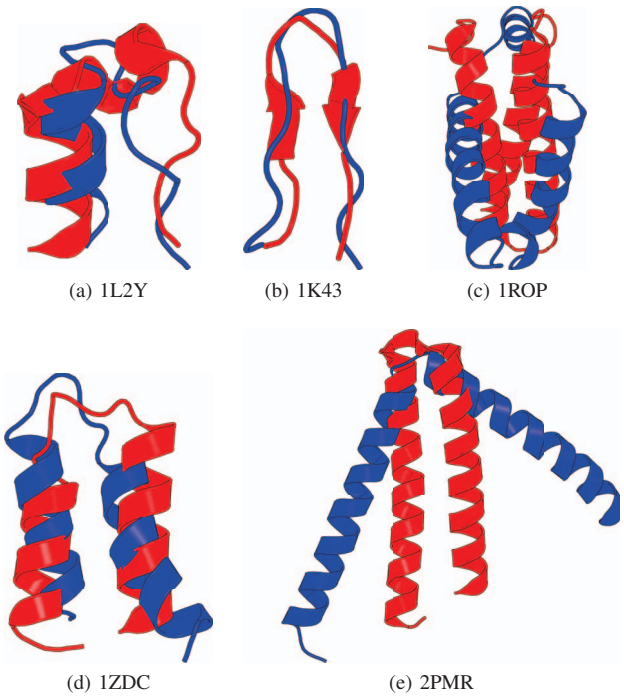
VI. Conclusion

As corroborated by CASP results achieved over the last years [18], there is an increasing need for new strategies for extracting, representing and manipulating structural data from experimentally determined 3-D protein structures, as well as for the development of computational strategies that make use of this information in order to predict the corresponding 3-D structure of proteins. As could be observed, Neuro evolution of augmenting topologies can be successfully applied to learn complex structural patterns from different protein data efficiently, without the need of previous knowledge for the ANNs design. This knownledge can be further used to reduce the protein data space in search strategies.

There is ample room for further development of the proposed methods: a) test the proposed method to other classes of proteins; b) create a database of trained neural networks;

TABLE IV: Secondary Structure content of predicted (P) and experimental (E) three-dimensional structures.

Protein	Strand P (E) %	α-helix P (E) %	3^{10}-helix P (E) %	Other P (E) %	Residues	RMSD Å
1L2Y	0.0 (0.0)	35.0 (35.0)	0.0 (20.0)	65.0 (45.0)	20	3.4
1K43	0.0 (42.9)	0.0 (0.0)	0.0 (0.0)	100.0 (57.1)	14	1.7
1ROP	0.0 (0.0)	92.9 (89.3)	0.0 (0.0)	7.1 (10.7)	56	11.6
1ZDC	0.0 (0.0)	76.5 (73.5)	0.0 (00.0)	23.5 (26.5)	34	3.6
2PMR	0.0 (0.0)	75.0 (80.3)	0.0 (0.0)	25.0 (19.7)	76	19.0



(a) 1L2Y (b) 1K43 (c) 1ROP

(d) 1ZDC (e) 2PMR

Fig. 7: Experimental (red) and predicted structures (blue). The C_α of the experimental and predicted structures are fitted. Graphics prepared with PYMOL.

c) examine the effect of the different size of amino acid fragments; d) use search methods such as GA, PSO, Simulated Annealing, GRASP or TABU search, to search the 3D protein conformational space using the computed intervals; e) test different metaheuristics to evolve the topology of ANNs; and f) consider more structural information to the training task.

ACKNOWLEDGMENT

This work was partially supported by grants from FAPERGS (002021-25.51/13) and MCT/CNPq (473692/2013-9 and 311022/2015-4), Brazil.

REFERENCES

[1] Structural mimicry of a native protein by a minimized binding domain. *Proc. Natl. Acad. Sci. USA*, 94(19).

[2] S.F. Altschul, W. Gish, W. Miller, E.W. Myers, and D.J. Lipman. Basic local alignment search tool. *J. Mol. Biol.*, 215(3):403–410, 1990.

[3] Jose Mariano Amabis and Gilberto Rodrigues Martho. *Fundamentos da Biologia Moderna*. Moderna, Sao Paulo, Brazil, 4 edition, 2006.

[4] C.B. Anfinsen. Principles that govern the folding of protein chains. *Science*, 181(96):223–230, 1973.

[5] D.W. Banner, M. Kokkinidis, and D. Tsernoglou. Structure of the cole1 rop protein at 1.7 a resolution. *J. Mol. Biol.*, 196:657–675, 1987.

[6] J. U. Bowie, R. Luthy, and D. Eisenberg. A method to identify protein sequences that fold into a known three-dimensional structure. *Science*, 253(5016):164–170, 1991.

[7] Peter Chervenski. Multineat, 2012–. Online; accessed 2016-01-28.

[8] P. Crescenzi, D. Goldman, C.H. Papadimitriou, A. Piccolboni, and M. Yannakakis. On the complexity of protein folding. *J. Comput. Biol.*, 5(3):423–466, 1998.

[9] M. Dorn, M. Barbachan e Silva, L. S. Buriol, and L. C. Lamb. Three-dimensional protein structure prediction: Methods and computational strategies. *Comp. Biol. and Chem.*, 53, Part B:251 – 276, 2014.

[10] Márcio Dorn, Luciana S. Buriol, and Luis C. Lamb. Moirae: A computational strategy to extract and represent structural information from experimental protein templates. *Soft Computing*, 18(4):773–795, 2013.

[11] J. Fauchere and V. Pliska. Hydrophobic parameters pi of amino-acid side-chains from the partitioning of n-acetyl-amino-acid amides. *Eur. J. Med. Chem.*, 18:369–375, 1983.

[12] M. Heinig and D. Frishman. Stride: a web server for secondary structure assignment from known atomic coordinates of proteins. *Nucleic Acids Res.*, 32(Web Server issue):W500–2, 2004.

[13] T.Z. Hovmoller and T. Ohlson. Conformation of amino acids in protein. *Acta Crystallogr.*, 58(5):768–776, 2002.

[14] Yin-Fu Huang and Shu-Ying Chen. Extracting physicochemical features to predict protein secondary structure. *The Scientific World Journal*, 2013, 2013.

[15] E.G. Hutchinson and J.M. Thornton. Promotif: A program to identify and analyze structural motifs in proteins. *Protein Sci.*, 5(2):212–220, 1996.

[16] SC Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(2):241–254, 1966.

[17] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. Online; accessed 2016-01-28.

[18] A. Kryshtafovych, K. Fidelis, and J. Moult. Casp10 results compared to those of previous casp experiments. *Proteins: Structure, Function, and Bioinformatics*, 82:164–174, 2014.

[19] R. A. Laskowski, G. E. Hutchinson, A. D. Michie, A. C. Wallace, M. L. Jones, and J. M. Thornton. PDBsum: a web-based database of summaries and analyses of all PDB structures. *Trends Biochem.Sci.*, 22(12):488–490, 1997.

[20] A.L. Lehninger, D.L. Nelson, and M.M. Cox. *Principles of Biochemistry*. W.H. Freeman, New York, USA, 4 edition, 2005.

[21] M.A. Martí-Renom, A. Stuart, A. Fiser, A. Sanchez, F. Mello, and A. Sali. Comparative protein structure modeling of genes and genomes. *Annu. Rev. Biophys. Biomol. Struct.*, 29(16):291–325, 2000.

[22] J.W. Neidigh, R.M. Fesinmeyer, and N.H. Andersen. Designing a 20-residue protein. *Nat.Struct.Biol.*, 9:425–430, 2002.

[23] D.J. Osguthorpe. Ab initio protein folding. *Curr. Opin. Struct. Biol.*, 10(2):146–152, 2000.

[24] M. T. Pastor, M. Lopez de la Paz, E. Lacroix, L. Serrano, and E. Perez-Paya. Combinatorial approaches: A new tool to search for highly structured -hairpin peptides. *Proc.Natl.Acad.Sci.USA*, 99(2):614–619, 2002.

[25] C.A. Rohl, C.E. Strauss, K.M.S. Misura, and D. Baker. Protein structure prediction using rosetta. *Methods Enzymol.*, 383(2):66–93, 2004.

[26] S. Sohangir, S. Rahimi, and B. Gupta. Neuroevolutionary feature selection using neat. *Journal of Software Engineering and Applications*, 7(7):562–570, 2014.

[27] Kenneth O. Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *The MIT Press Journals Evolutionary Computation*, 10(2):99–127, 2002.

[28] Sheng Wang, Jian Peng, Jianzhu Ma, and Jinbo Xu. Protein secondary structure prediction using deep convolutional neural fields. *ArXiv e-prints*, 2015.