Protein Structural Blocks Representation and Search through Unsupervised NN

Virginio Cantoni[1], Alessio Ferone[2], Ozlem Ozbudak[3], and Alfredo Petrosino[2]

[1] University of Pavia, Department of Electrical and Computer Engineering, Via A. Ferrata, 1, 27100, Pavia, Italy
virginio.cantoni@unipv.it
[2] University of Naples Parthenope, Department of Applied Science, Centro Direzionale Napoli - Isola C4, 80143, Napoli, Italy
{alessio.ferone,alfredo.petrosino}@uniparthenope.it
[3] Istanbul Technical University, Department of Electronics and Communication Engineering, Ayazaga Campus, 34469, Maslak, Istanbul, Turkey
ozbudak@itu.edu.tr

Abstract. This paper aims to develop methods for protein structural representation and to implement structural blocks retrieval into a macromolecule or even into the entire protein data-base (PDB). A first problem to deal with is that of representing structural blocks. A proposal is to exploit the Extended Gaussian Image (EGI) which maps on the unitary sphere the histogram of the orientations of the object surface. In fact, we propose to adopt a particular 'abstract' data-structure named Protein Gaussian Image (PGI) for representing the orientation of the protein secondary structures (helices and sheets). The 'concrete' data structure is the same as for the EGI, however, in this case the points of the Gaussian sphere surface do not contain the area of patches having that orientation but features of the secondary structures (SSs) having that direction. Among the features we may include the versus (e.g. + origin versus surface or - vice versa), the length of the structure (number of amino), biochemical properties, and even the sequence of the amino (such as in a list). We consider this representation very effective for a preliminary screening when searching on the PDB. We propose to employ the PGI in a fold recognition problem in which the learning task is performed by means an unsupervised method for structured data.

1 Introduction

There are currently 80,041 (at 13/3/2012) experimentally determined 3D structures of protein deposited in the Protein Data Bank (PDB) [1] (with an increment of about 700 new molecules for month). However this set contains a lot of very similar (if not identical) structures. The importance of the study of structural building blocks, their comparison and their classification, is instrumental to the study on evolution and on functional annotation, and has brought about many methods for their identification and classification in proteins of known

A.E.P. Villa et al. (Eds.): ICANN 2012, Part II, LNCS 7553, pp. 515–522, 2012.

structure. The procedures, often automatic or semi-automatic, for reliable assignment are essential for the generation of the databases (especially as the number of protein structures is increasing continuously) and the reliability and precision of the taxonomy is a very critical subject. This also because there is no standard definition of what a structural motif, a domain, a family, a fold, a sub-unit, a class, etc. really is, so that assignments have varied enormously, with each researcher (other than for each DB) using a its own set of criteria.

There are several DBs for structural classification of proteins; among them the most commonly used are Structural Classification Of Proteins (SCOP) [13] and Class Architecture Topology and Homologous super families (CATH) [14]. The problem of structural representation of a macromolecule until now has been pursued by 'ad hoc' descriptors of patterns which are often point-based and cumbersome for the management and processing. Among these we can quote: spin image [5], [6], context shape [7] and harmonic shape [8]. For our purposes we intend to propose a new approach based on an appropriate extended Gaussian image (EGI) [9] which represents the histogram of the surface orientations. The EGI, introduced, for applications of photometry, by B.K.P. Horn [9], has been extended by K. Ikeuci [10] [11] [12] (the Complex-EGI). Despite the many variants of the Gaussian Image, to our knowledge, this type of representation has never been applied to proteins; we intend to present a new version, suitable for this context and to employ it in a structural recognition task [4]. In particular, the PGI will be used in a fold recognition problem in which the learning task is performed by SOM–SD, a self organizing map for structured data.

The remainder of the paper is as follows. In Section 2 the Protein Gaussian Image is explained. In Section 3 a brief overview of the SOM–SD method is given. Section 4 shows preliminary experimental results and Section 5 concludes the paper.

2 The Protein Gaussian Image

There are several methods for defining protein secondary structure, but the Dictionary of Protein Secondary Structure (DSSP) [2] approach is the most commonly used. The DSSP defines eight types of secondary structures, nevertheless, the majority of secondary prediction methods simplify further to the three dominant states: Helix, Sheet and Coil. Namely, the helices include 3/10 helix, α–helix and π–helix; sheets or strands include extended strand (in parallel and/or anti-parallel β–sheet conformation); finally, coils which can be considered just connections. In the sequel, the structural analysis for protein recognition and comparison, is conducted only on the basis of the two most frequent components [3]: the α–helices and the β–strands. DSSP and STRIDE both extract from the PDB segments 3D locations and attitudes, positions in the sequence of SSs and in particular also strands (constituting sheets), and many other information easily integrated in the new data structure.

PGI is a representation in the Gaussian image in which each SS is mapped with a unit vector from the origin of the sphere having the orientation of the SS. Each point of the sphere surface contains the data orientation (length, location of starting and ending residue, etc) of the existing protein SSs having the corresponding orientation. The chain sequence of SS is recorded as a list which is mapped on the sphere surface.

The proposed data structure is complete (no information is lost for an analytic analysis) and effective from the computational viewpoints (only two reference coordinates are needed), but also, as for needle maps for general object representation, supports effectively the structural perception. In order to validate the effectiveness of the PGI representation of the protein structure, we propose to employ an unsupervised framework for structured data in a pratical structural learning problem, where each protein is represented by a PGI.

Before showing the results, we give two practical examples of how proteins are represented employing the PGI data–structure. Two couples of motif-protein molecules are shown: 1FNB and 4GCR. For the former couple Figure 1(a) represents the molecule, Figure 1(b) shows the PGI (in green the Greek key motif is highligthed) while Figure 1(c) shows the PGI of the motif with the linked sequence of SSs. In Figures 2(a)-2(c) the analogous representation for the latter couple are shown.

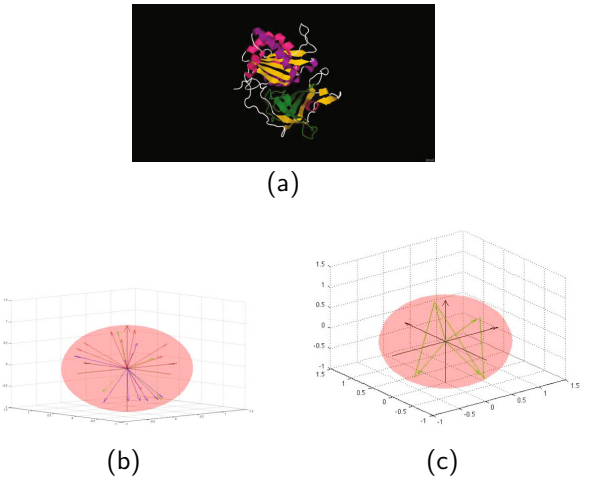

Fig. 1. 1(a) A picture generated by PyMOL on PDB file 1FNB rotated $\pi/2$ for format reasons. In green the Greek key motif (residues 56-116). 1(b) Protein Gaussian Image of protein 1FNB. Green arrows represent the Greek key motif. 1(c) Protein Gaussian Image of Greek motif contained in protein 1FNB. Green arrows represent the Greek key motif, while the green line shows the sequence of SS.

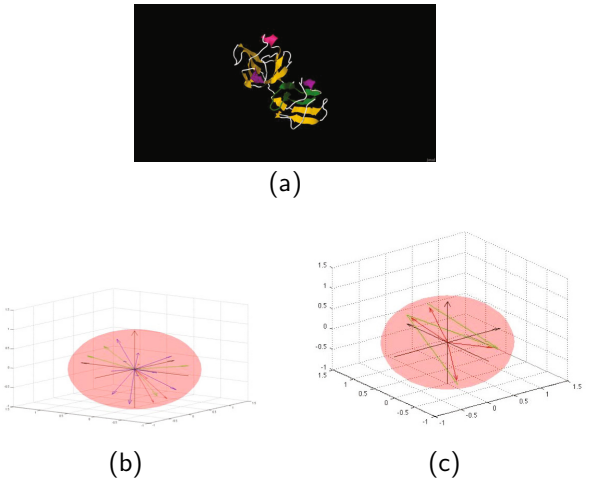

Fig. 2. 2(a) A picture generated by PyMOL on PDB file 4GCR rotated $\pi/2$ for format reasons. In green the Greek key motif (residues 34-62). 2(b) Protein Gaussian Image of protein 4GCR. Green arrows represent the Greek key motif. 2(c) Protein Gaussian Image of Greek motif contained in protein 4GCR. Green arrows represent the Greek key motif, while the green line shows the sequence of SS.

3 Learning Structured Data

In the recent years, complex data structures have been employed to obtain a better representation of data and possibly a better solution for a given problem. Many methods are based on a preprocessing step that first maps the strucutred data to a vector of reals which then can be processed in the usual manner. However, important information, like topological relationship between nodes, may be lost during the preprocessing phase, so that the final result strongly depends on the adopted algorithm. Neural methods are an example of techniques that evolved to handle structured data where the original connectionist models have been modified to process different data structures.

In order to prove the effeciveness of the PGI, we propose to perform a learning task by employing the SOM–SD framework [15], an unsupervised method for structured data. Throughout this section we will use the same notation as in [15].

3.1 SOM Framework for Structured Data

The SOM–SD framework can be defined by using a computational framework similar to that defined for recursive neural networks (RNN) [16]. The class of functions which can be realized by a RNN can be characterized as the class of functional DAG (Directed Acyclic Graph) transductions $\tau : \mathcal{I}^{\#} \rightarrow \mathbb{R}^k$, which can be represented in the following form $\tau = g \circ \hat{\tau}$, where $\hat{\tau} : \mathcal{I}^{\#} \rightarrow \mathbb{R}^n$ is the encoding function and $g : \mathbb{R}^n \rightarrow \mathbb{R}^k$ is the output function. $\hat{\tau}$ is defined recursively as:

$$\hat{\tau}(\mathcal{D}) = \begin{cases} \mathbf{0}(\text{the null vector in } \mathbb{R}^n), & if \mathcal{D} = \text{void graph} \\ \\ \tau(\mathbf{y}_s, \hat{\tau}(\mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(c)})), & otherwise \end{cases} \tag{1}$$

where $\mathcal{D}$ is a DAG, $\mathbf{y}_s$ is the label of the supersource of $\mathcal{D}$ and τ is defined as:

$$\tau : \mathbb{R}^m \times \underbrace{\mathbb{R}^n \times \ldots \times \mathbb{R}^n}_{c \text{ times}} \to \mathbb{R}^n \tag{2}$$

A typical form for τ is:

$$\tau(\mathbf{u}_v, \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(c)}) = \mathbf{F}(\mathbf{W}\mathbf{u}_v + \sum_{j=1}^{c} \widehat{\mathbf{W}}_j \mathbf{x}^j + \theta) \tag{3}$$

where $\mathbf{F}_i(\mathbf{v}) = sgd(v_i)$ (sigmoidal function), $\mathbf{u}_v \in \mathbb{R}^m$ is a label, $\theta \in \mathbb{R}^n$ is the bias vector, $\mathbf{W} \in \mathbb{R}^{m \times n}$ is the weight matrix associated with the label space, $\mathbf{x}^{(j)} \in \mathbb{R}^m$ are the vectorial codes obtained by the application of the encoding function $\hat{\tau}$ to the subgraphs $\mathcal{D}^{(j)}$ (i.e., $\mathbf{x}^{(j)} = \hat{\tau}(\mathcal{D}^{(j)})$), and $\widehat{\mathbf{W}}_j \in \mathbb{R}^{m \times m}$ is the weight matrix associated with the j–th subgraph space. The output function g is generally realized by a feedforward neural network (NN).

The key consideration to adapt this framework to the unsupervised SOM approach [17], is that the function τ maps information about a node and its children from a higher dimensional space (i.e., $m + c \cdot n$) to a lower space (i.e., n). The aim of the SOM learning algorithm is to learn a feature map

$$\mathcal{M} : \mathcal{I} \to \mathcal{A} \tag{4}$$

which, given a vector in the input space $\mathcal{I}$ returns a point in the output space $\mathcal{A}$. This is obtained in the SOM by associating each point in $\mathcal{A}$ to a different neuron. Given an input vector v, the SOM returns the coordinates within $\mathcal{A}$ of the neuron with the closest weight vector. Thus, the set of neurons induces a partition of the input space $\mathcal{I}$. In typical applications $\mathcal{I} \equiv \mathbb{R}^m$, where $m \gg 2$, and $\mathcal{A}$ is given by a two dimensional lattice of neurons. In this way, input vectors which are close to each other will activate neighbor neurons in the lattice. SOM–SD represents an extension of the SOM framework, where the τ function is implemented in an unsupervised learning framework, in order to generalize 4 to deal with the case $\mathcal{I} \equiv \mathcal{Y}^{\#^{(c)}}$, i.e., the input space is a structured domain with labels in $\mathcal{Y}$. The function

$$\mathcal{M}^{\#} : \mathcal{Y}^{\#^{(c)}} \to \mathcal{A} \tag{5}$$

is realized by defining 1 as shown in 6

$$\mathcal{M}^{\#}(\mathcal{D}) = \begin{cases} nil_{\mathcal{A}}, & \text{if } \mathcal{D} = \text{void graph} \\ \\ \mathcal{M}_{node}(\mathbf{y}_s, \mathcal{M}^{\#}(\mathcal{D}^{(1)}), \ldots, \mathcal{M}^{\#}(\mathcal{D}^{(c)})), & otherwise \end{cases} \tag{6}$$

where $s = source(\mathcal{D})$, $\mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(c)}$ are the subgraphs pointed by the outgoing edges leaving from s, $nil_{\mathcal{A}}$ is a special coordinate vector into the discrete output space $\mathcal{A}$, and

$$\mathcal{M}_{node} : \mathcal{Y} \times \underbrace{\mathcal{A} \times \ldots \times \mathcal{A}}_{c \text{ times}} \to \mathcal{A} \tag{7}$$

is a SOM, defined on a generic node, which takes as input the label of the node and the "encoding" of the subgraphs $\mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(c)}$ according to the $\mathcal{M}^{\#}$ map. By "unfolding" the recursive definition in 6, it turns out that $\mathcal{M}^{\#}(\mathcal{D})$ can be computed by starting to apply $\mathcal{M}_{node}$ to leaf nodes (i.e., nodes with null outdegree), and proceeding with the application of $\mathcal{M}_{node}$ bottom-up from the frontier nodes (sink nodes) to the supersource of the graph $\mathcal{D}$.

Details about the learning algorithm can be found in [15].

4 Experimental Results

This section shows preliminary results obtained by SOM–SD using as input a set of proteins represented as PGI. In particular, for each secondary structure, only its direction is considered. The dataset is composed of 45 proteins classified by SCOP as belonging to the class "Alpha and beta proteins (a/b)". Three folds have been considered, namely "Flavodoxin–like" "RibonucleaseH-likemotif" and "TIMbeta/alpha-barrel", and for each fold 15 proteins have been chosen. The task consists in grouping proteins or side chains belonging to the same fold.

The network performances will be reported in terms of *clustering performance*, *classification performance*, and *retrieval performance*. The clustering performance is a measure on how well clusters are formed. This measure does not take into account the desired clustering, not considering the target labels. The classification performance is a measure on how well the clustering corresponds to the desired clustering by comparing the target values of vertices mapped to the same location. The retrieval performance is a measure of confidence of the clustering result, so that if there are many vertices with different targets mapped to the same location, the confidence in the clustering is low, hence producing a low retrieval performance. The SOM–SD was trained on 200×200 neurons with a neighborhood spread of $\sigma = 60$, considering different learning rate $\eta = [1$ 1.25 $1.5]$ and differents iterations $\lambda = [40$ 60 $80]$. For each test the average performance over 50 runs is reported. The dataset for each test was composed by randomly picking the 70% of the patterns for the training phase and the remaining 30% for the testing phase.

The first test has been conducted considering the whole protein as pattern, i.e., each protein is represented by a PGI. It can be observerd in Table 1 that the results are quite good in term of clustering performance. Even though this measure does not take into account the desired clustering outcome, the result is supported by the good retrieval performance which reflects a reduced confusion in the mappings of each pattern. The classification performance, reflecting the

Table 1. Performance of a 200 × 200 SOM–SD considering the whole proteins

	Test Set								
Learning Rate	1			1.25			1.5		
Iterations	40	60	80	40	60	80	40	60	80
Retrieval	84,99	84,08	84,86	86,46	87,69	79,79	79,29	82,31	79,82
Classification	72,65	56,95	56,91	59,39	70,65	60,73	65,91	64,30	74,82
Clustering	0,79	0,85	0,83	0,82	0,81	0,85	0,83	0,79	0,80

performance with respect to the desired clustering outcome, shows less accurate results, but with an interesting peak at $74,82\%$

The second test has been performed considering as patterns the single side chains of each protein, i.e., each side chain is represented by a PGI. From Table 2 it can be noted how this "reduced" representation yields better results in term of classification and retrieval performance while performing slightly worst with respect to clustering performance. In particular, the clustering performance is almost the same but with a higher confidence reflected by the higher retrieval performance. The interesting result concerns the classification performance that is much higher considering only the side chain.

Table 2. Performance of a 200 × 200 SOM–SD considering the single side chains of each protein

	Test Set								
Learning Rate	1			1.25			1.5		
Iterations	40	60	80	40	60	80	40	60	80
Retrieval	74,39	81,67	79,63	92,72	92,35	93,40	92,36	92,34	93,85
Classification	75,58	76,37	77,64	85,11	84,14	84,17	85,03	85,78	86,42
Clustering	0,8	0,80	0,80	0,79	0,80	0,79	0,79	0,80	0,79

5 Conclusion

A new data structure has been introduced that supports both artificial and human analysis of protein structure. Preliminary tests have been performed by employing the PGI in a structural learning task obtaining very good and promising results. We are now planning an intensive quantitative analysis of the effectiveness of this new representation approach for practical problems such as alignment or even of structural block retrieval at different level of complexity: from basic motifs composed of a few SSs, to domains, up to units. Moreover, in future works we will try to exploit other feature beside the orientation, like the length of the structure in terms of number of amino, biochemical properties, the sequence of the amino, etc.

References

1. Protein Data Bank, http://www.pdb.org/
2. Kabsch, W., Sander, C.: Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. Biopolymers 22(12), 2577–2637 (1983)
3. David, E.: The discovery of the alfa-helix and beta-sheet, the principal structural features of proteins. Proceedings of the National Academy of Sciences USA, 100:11207–100:11210 (2003)
4. Cantoni, V., Ferone, A., Petrosino, A.: Protein Gaussian Image (PGI) - A Protein Structural Representation Based on the Spatial Attitude of Secondary Structure. New Tools and Methods for Pattern Recognition in Complex Biological Systems (in press)
5. Shulman-Peleg, A., Nussinov, R., Wolfson, H.: Recognition of Functional Sites in Protein Structures. J. Mol. Biol. 339, 607–633 (2004)
6. Bock, M.E., Garutti, C., Guerra, C.: Spin image profile: a geometric descriptor for identifying and matching protein cavities. In: Proc. of CSB, San Diego (2007)
7. Frome, A., Huber, D., Kolluri, R., Bülow, T., Malik, J.: Recognizing Objects in Range Data Using Regional Point Descriptors. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3023, pp. 224–237. Springer, Heidelberg (2004)
8. Glaser, F., Morris, R.J., Najmanovich, R.J., Laskowski, R.A., Thornton, J.M.: A Method for Localizing Ligand Binding Pockets in Protein Structures. PROTEINS: Structure, Function, and Bioinformatics 62, 479–488 (2006)
9. Horn, B.K.P.: Extended Gaussian images. Proc. IEEE 72(12), 1671–1686 (1984)
10. Kang, S.B., Ikeuchi, K.: The complex EGI: a new representation for 3-D pose determination. In: IEEE-T-PAMI, pp. 707–721 (1993)
11. Shum, H., Hebert, M., Ikeuchi, K.: On 3D shape similarity. In: Proceedings of the IEEE-CVPR 1996, pp. 526–531 (1996)
12. Kang, S., Ikeuchi, K.: The complex EGI, a new representation for 3D pose determination. IEEE Trans. Pattern Anal. Mach. Intell. 15(7), 707–721 (1993)
13. Murzin, A.G., Brenner, S.E., Hubbard, T., Chothia, C.: SCOP: a structural classification of proteins database for the investigation of sequences and structures. J. Mol. Biol. 247, 536–540 (1995)
14. Orengo, C.A., Michie, A.D., Jones, S., Jones, D.T., Swindells, M.B., Thornton, J.M.: CATH–a hierarchic classification of protein domain structures. Structure 5(8), 1093–1108 (1997)
15. Hagenbuchner, M., Sperduti, A., Tsoi, A.H.: A self-organizing map for adaptive processing of structured data. IEEE Transactions on Neural Networks 14(3), 491–505 (2003)
16. Sperduti, A.: Knowledge-Base Neurocomputing. In: Cloete, I., Zurada, J.M. (eds.) pp. 117–152. MIT Press, Cambridge (2000)
17. Kohonen, T.: Self–Organization andAssociative Memory, 3rd edn. Springer, New York (1990)