Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: 21 NOV 2019
Manuscript Category

OXFORD

A comparison between GOR method and Support Vector Machines.

Predicting secondary structure of proteins

Lui Maria [1]

[1] Department of Pharmacy and Biotechnology, University of Bologna, Bologna,40126,Italy

Abstract

Motivation: Nowadays the investigation on protein properties is the biggest challenge in the bioinformatic field in order to understand and explain their functions. In particular, the study of specific and organized interactions in space and time between proteins, lipids, nucleic acids and the cell metabolites requires some knowledge about how a protein can fold in a unique and active three-dimensional structure. Because of the big accumulation of protein sequences and, on the other hand, the slower experimental determination of protein structures, the use of methods for protein secondary structure predictions becomes crucial. Therefore we tested two different methods for the secondary structure prediction : the GOR method and the SVM method, the first based on Bayesian statistic and the second on machine learning.
Results: From the analysis of the performance that we carried out for both the methods, becomes apparent that SVM performs a better overall prediction, more accurate and more precise than the one of GOR. Despite this, the SVM method showed an high sensitivity in predicting helix and coil conformations but the strand values were estimated quite low.
Contact: maria.lui@studio.unibo.it
Supplementary information: Supplementary files available at *https://github.com/marialui/LB2* online or in the attached PDF

1 Introduction

Proteins mediate the largest amount of functions inside the cell; they give rise to an high number of interactions (so called PPI network) involved into the metabolism or signal transduction, they can act as enzymes, they can have a structural role, regulation of genetic expression, or they can be involved in the transport of molecules. The shape of a protein is very important to perform each of these functions.
We can define a first level of organization of a protein called primary structure: it consists on a sequence of amino acids in a polypeptide chain. But the primary structure per se, doesn't provide information about solvent-accessible surface area, the shape of a protein, surfaces characteristics and residue interface propensities.[Jones and Thornton, 1996] Moreover, when a protein loses its higher-order structure, even though the primary sequence doesn't change, those proteins are usually non-functional. Thus, in this context, the study of the molecular structure gains in importance. The second level of protein organization, called secondary structure, refers to local folded structures formed within a polypeptide due to the interactions between atoms of the backbone. Indeed, amino acids vary in their ability to form different stable, short-range, periodic folding elements. Those folding elements define a specific secondary structure and are characterized by dihedral angles ψ and ϕ that remain the same or nearly the same.
There are few types of secondary structures that occur more often in proteins. The most common are α helices, β sheets and besides them there are random coil. The last ones are not properly secondary structure elements but refer to the absence of any secondary structure.
The third level of organization is called tertiary structure and refers to the specific shape acquired by a polypeptide chain in the three-dimensional space. It is mainly due to side-chain interactions of the type charge-charge,to hydrophobic interactions, disulfide bonds and Van der Waals forces. [Engelking, 2014] Nowadays a huge amount of data is available for the primary structure of proteins. But only a few of these has been experimentally annotated with secondary and tertiary structure. Since to perform a specific function a precise protein shape is crucial, new bioinformatic methods have been developed in order to predict new structures starting from protein sequences. Between the 1960s and the 1970s, the so called first generation prediction methods have been developed, based on single amino acid propensities and either simple stereochemical principles [Lim, 1974] or statistics [Chou and Fasman, 1974] to identify local secondary structure motifs. This class of methods reached an accuracy of about 60% . A second generation of methods have been spread out between 1970s and the 1990s, with the evaluation of propensities for segments of 3-51 adjacent residues, so that the improvement was in the evaluation of the context of each single position (e.g.: GOR [Garnier *et al.*, 1978]). The use of this second class of methods improved the levels of accuracy above 60%. The main improvements came in the 1990s from the use of multiple sequence alignments that led to a better overall accuracy for structure prediction (levels above 70%). Those methods rely on a combination of large databases and advanced algorithms

([Rost *et al.*, 1994] Here we analyse in detail two of these methods: the GOR method (based on simple statistical analysis) and the Support Vector Machine (which is based Machine learning). At first we endowed a set of proteins with their predicted secondary structure, throughout the use of both these methods. Then, we compared the results with the DSSP assignments of secondary structure deriving from a pattern-recognition process of hydrogen-bonded and geometrical features extracted from x-ray coordinates . [Kabsch and Sander, 1983]. Finally we computed the performance for both the GOR and SVM in order to find out which is the best approach for the prediction of protein secondary structure.

2 Materials and methods

In order to train the models and later on to evaluate both the two methods we used two different sets:

- A training set (The JPred4 dataset)
- A Blind-test set

2.1 Training set

The training set, the JPred4 dataset, has been produced starting from 1987 single representative sequences of each super family in SCOPe v.2.04.[Drozdetskiy *et al.*, 2015] Then low resolution structures (> 2.5 Å) have been removed and sequences filtered out for their length (< 30 and > 800 residues in length). Another step was the deletion of multi-chain or fragment domains and sequences with missing DSSP information from the training set. Then all those structures showing inconsistencies between PDB, DSSP and ASTRAL file have been excluded from the set. Finally, authors split data into a training and blind test. But, for our purpose, we used only the training portion made up of 1348 structures.

2.2 Blind test set

We produced a Blind test set in order to evaluate the GOR and SVM models. Firstly, we downloaded from PDB [RCSPDB, 2019] all structures meeting the following pre-defined criteria:

- Experimental method: X-ray cristallography
- Resolution < 2.5 Å
- Release date after Jan, 2015
- Chain length between 50 and 300 residues
- Internal dataset redundancy has been reduced to 30%

Then, we used a python script to extract all chain sequences in Fasta format, removing identical ones. In case of multiple chains of the same length, we retained the one with the best resolution. We further reduce the internal redundancy of the dataset using BLASTCLUST [Dondoshansky and Wolf, 2002] that automatically and systematically clusters protein sequences based on pairwise matches found using the BLAST algorithm. It uses a FASTA-format sequence file as input and produces clusters of sequence IDs. Sequences will be clustered together under a specific threshold of :

- L = minimum length coverage
- S = sequence identity

We decided to remove the redundancy with a threshold 30% of sequence identity. From this set of clusters, we extracted just one representative, the one solved with the best resolution; so we finally ended up with 3016 sequences. Another step we did to build the blind set was to reduce the external redundancy to 30% sequence identity with respect to the JPred4 dataset using the command-line version of BLAST [Altschul *et al.*, 1990]. In order to make the software run faster, BLAST needs to do some pre-work on the database file: Before running BLAST we indexed the blind set using the makeblastdb tool. We used blastp, which is appropriate for protein to protein comparisons, to search for similar sequences, specifying the input query sequences (jpred multifasta), the database (indexed blind test file), the E-value threshold (0.01), used to filter out less significant results, and ask blastp to provide tabular output. From the result of this blastsearch, proteins with a sequence identity bigger than 30% with respect to the jpred training set, have been filtered out (473 sequences removed). The last step was the selection of 150 random proteins from the obtained dataset. Once all the PDB files for the 150 ids of the blind test set have been downloaded, we run mkdssp, a program that calculates the most likely secondary structure assignment given the 3D structure of a protein. In particular we obtained only 147 DSSP because mkdssp was not able to predict the secondary structure for: 5MKL, 4YBB, 4YBB. DSSP outputs include 8 different secondary structure conformations that have been mapped into three classes as follows:

- α -helix (H),3-helix (310 helix (G) and 5 helix (π -helix (I) has been mapped to helix (H)
- Residue in isolated β-bridge (B),extended strand, participates in β ladder (E) has been mapped to strand (E)
- Hydrogen bonded turn (T), bend (S) and positions with no assignment ("") has been mapped to coil (C)

2.3 Statistical analysis of the datasets

A basic statistic analysis was carried out for both the training and the blind test set in order to assess their suitability. In particular we evaluated the distribution of secondary structure conformations (Figure 4), the residue and structural composition (Figure 1), taxonomic classification for kingdom (Figure 2) and species (Figure 3), and structural classification (Figure 5). This statistical analysis of the datasets didn't reveal significant differences between the training and the blind test set.

2.4 Generating sequence profiles

We can study structurally or functionally important residues of a protein looking at conserved positions in a multiple sequence alignment. Therefore, we generated a set of sequence profiles parsing the output of PSIBLAST (Position-Specific Iterative Basic Local Alignment Search Tool). A sequence profile is a compact representation of a protein family specifically built for a target protein sequence. It is produced converting multiple sequence alignments into position-specific scoring systems (PSSMs). Each of the 20 amino acids in each position of the alignment are scored according to the frequency in which they occur in the original alignment [Gribskov *et al.*, 1987].
We launched PSIBLAST for each file of both the blind test and the training set against the entire UniProtKB/SwissProt database [UniProt, 2019] so that it derives a position-specific scoring matrix (PSSM) from the multiple sequence alignment of sequences detected above score threshold of 0.01. This PSSM is used to further search the database for new matches, and is updated for 3 subsequent iterations with these newly detected sequences.[Altschul *et al.*, 1997] Once we obtained a PSSM file for each protein we run a python script to extract the sequence profile and we normalize each value by dividing for 100. Some of those profiles resulted to be full of zeros so we decide to remove them and we ended up with 1224 profiles for the training set and 120 profiles for the blind test set.

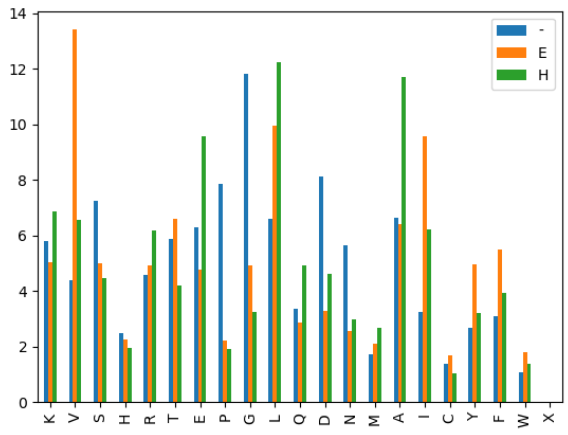

Fig. 1. Residue and structures composition.

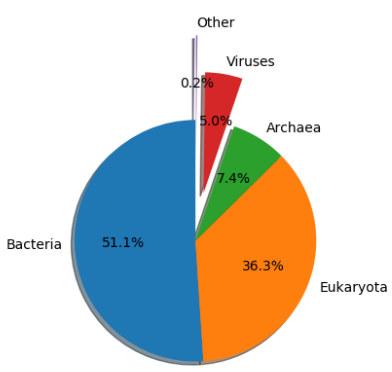

Fig. 2. Kindom classification.

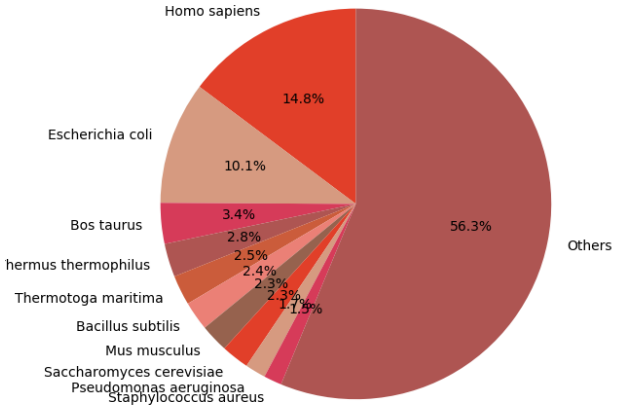

Fig. 3. Taxonomic classification.

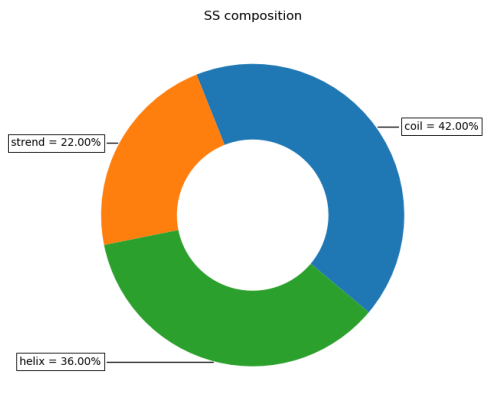

Fig. 4. structures composition.

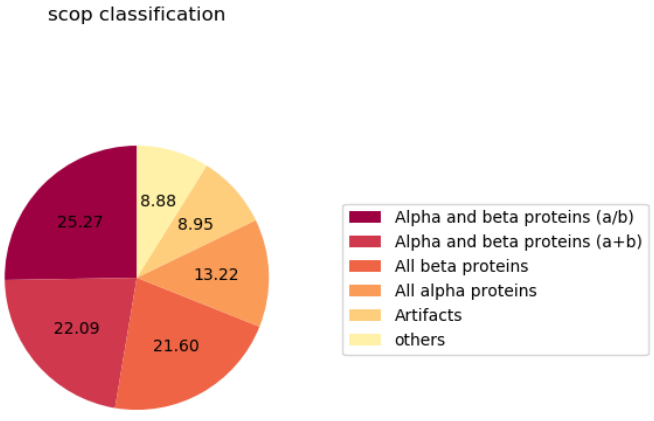

Fig. 5. Scop classification.

2.5 GOR method

The GOR (Garnier–Osguthorpe–Robson) is one of the most popular methods for secondary structure prediction; it uses both information theory and Bayesian statistics to predict the secondary structure starting from an amino acid sequence. Its main improvement consist in the evaluation of the context for each position of the sequence. In fact it relies on the computation of an Information Function (Equation 1) extended and defined over windows of residues that ranges from 8 residues before and 8 residue after a central position. This is possible thanks to the independence assumption that states the statistical independence contribution of context residues to the central residue conformation. So there must be no correlation between residues occurring at different positions in the window of 17 residues.

$$I(S;R) = \log \frac{P(S|R)}{P(S)} \tag{1}$$

where:

- S is one of the three conformations
- R is one of the 20 aminoacid residues
- $P(S|R)$ is the conditional probability for observing a conformation S when a residue R is present
- $P(S)$ is the probability of observing S

According to the definition of conditional probabilities, we can write :

$$P(S|R) = \frac{P(S,R)}{P(R)} \tag{2}$$

Given that the joint probability of observing the events S and R is :

$$P(S,R) = \frac{f(S,R)}{N} \tag{3}$$

And $P(R)$ $P(S)$ are respectively the probability of observing a residue R and the probability of observing a structure S, are :

$$P(R) = \frac{f(R)}{N} \tag{4}$$

$$P(S) = \frac{f(S)}{N} \tag{5}$$

where:

- N is the total number of amino acids in the database
- $f(S)$ is the total number of residues observed in the conformation S
- $f(R)$ is the total number of residues R
- $f(S,R)$ is the number of residues R observed in the conformation S

GOR calculates 3 different information functions, one for each secondary structure (helix, strand and coil), as the sum of individual single-residue functions relating residues in the window with the central-residue conformation. The highest value of the of the window-based information function Equation (1) for one of the conformations S will be the predicted conformation . Considering a 17-residue symmetric window centred at a given residue R at position j,we can compute the predicted conformation as follows:

$$S^* = \arg\max_s I(S; R_{-8}, .., R_8) \tag{6}$$

$$S^* = \arg\max_s \sum_{k=-8}^{k=8} I(S; R_k) \tag{7}$$

The GOR algorithm has been implemented in python language.

2.6 Support Vector Machines method

Support Vector Machines are computational models that allow to separate linearly separable classes with hyper-planes accommodating the largest margin. The separating hyper-plane should be as far away from the data of both classes as possible. So we have to identify a set of support vectors x_i (that are the closest points to the hyper-plane) defining the maximum margin hyper-plane as:

$$W^T x + b = 0 \tag{8}$$

Where W is a vector perpendicular to the hyper-plane, with:

$$W = \sum_{i=1}^{n} \alpha_i y_i x_i \tag{9}$$

and b determined on a support vector. We should maximize the margin m:

$$m = \frac{2}{||W||} \tag{10}$$

Let $x1, ..., xn$ be our data set and let $y_i \in 1, -1$ be the class label of x_i:

$$W \times X_i + b \geq +1 \tag{11}$$

for $y_{=}1$
And :

$$W \times X_i + b \leq -1 \tag{12}$$

for $y_i = -1$
The decision boundary can be found solving the following constrained optimization problem:
The minimization of

$$\frac{1}{||W||^2} \tag{13}$$

subject to :

$$y_i \times (W^T x + b) \geq 0 \tag{14}$$

for $\forall y_i$
Choosing the hyperplane which maximizes the margin means to optimize a quadratic function subject to linear constraints. That's why we introduce the maximization of the objective function of the dual problem.

$$\max \mathcal{L}(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i x_j \rangle \tag{15}$$

subject to:

- $\alpha_i \geq 0$
- $\sum_{i=1}^{n} \alpha_i y_i = 0$

If the training set is not linearly separable we can use a soft margin classification : Slack variables ξ_i can be added to allow misclassification of difficult or noisy examples, resulting margin called soft. The previous hard-margin formulation will be modified to incorporate slack variables:

$$\frac{1}{||W||^2} + C \sum_{i=1}^{n} \xi_i \tag{16}$$

subject to:

- $y_i \times (W^T x + b) \geq 1 - \xi_i$
- $\xi_i \geq 0, \forall i$

The Hyper-parameter C has been introduced in order to avoid over-fitting: it acts as trade-off between the necessity of minimizing the errors and maximizing the margin. With soft margins, the dual problem is identical

to separable case but here it is subject to $0 \leq \alpha_i \leq C$.
Another way to perform non-linearly separable classification is mapping data to a higher-dimensional space: The idea is to use a transformation in the feature space to make a non linear separable problem, linearly separable

$$x \Rightarrow \phi(x) \tag{17}$$

Here, we use the so called Kernels functions that allow to apply such transformation on the original dataset. The kernel can be written indeed as scalar products in some feature space.

$$K(x1, x2) = \phi(x1)\phi(x2) \tag{18}$$

Thus, applying the kernel we are actually solving the same problem but in another feature space without computing each single transformation explicitly.
For the purpose of this project we perform the SVM prediction using The LIBSVM package [Chang and Lin, 2011], an integrated software for support vector classification, that supports multi-class classification. LIBSVM is a library for Support Vector Machines (SVMs) consisting in two main sub-routines that are SVM-train and SVM-predict. The first one will take in input a training file and will produce a model file in output, SVM-predict instead will test the model taking in input a testing file and the model itself and giving the prediction file as output.
SVM requires that each data instance (for both the training and the prediction) is represented as a vector of real numbers, so we wrote a python script that produces input in a suitable way for the LIBSVM package. Specifically,we had to convert categorical attributes (helix, strand, and coil) into numeric data (class 1, 2 and 3). To train SVM problems, we had to specify some parameters:

- SVM type : multi-class classification
- Kernel type : RBF (radial basis function), because it is characterized by a low number hyperparameters which influence the complexity of model selection and it has fewer numerical difficulties.
- Gamma and Cost (γ,C): For our purpose we performed a grid search for C and γ in order to find the best hyperparametres. We tested values of 2 and 4 for the Cost and values of 0.5 and 2 for γ.

LIBSVM indeed provides a simple tool to check a grid of parameters. For each parameter setting, LIBSVM obtains cross-validation (CV) accuracy so that, at the end, the parameters with the highest CV accuracy are returned. We provided 2 possible values of C and γ with the grid space so that all grid points (C,γ) have been tried in order to find the one giving the highest CV accuracy. Finally we used the best parameters (γ= 0.5 and C=2) to train the whole training set and to generate the final model.

3 Results and discussion

3.1 Measures of the performance

The evaluation of the performance consist in the comparison of the real (obtained via DSSP program) and the predicted secondary structure. SS prediction is a multi-class classification problem with 3 classes. Thus, in order to measure the performance for both the methods, we ran a python script able to compute a 3-class-confusion matrix (3x3 matrix for calculating the performance of a prediction methods).
In order to evaluate the performance we compute:

- Multi-class accuracy, a measure of how many predictions are correct on the overall

  $$Q3 = \frac{p_{HH} + p_{EE} + p_{CC}}{N} \tag{19}$$

  Where PHH, PEE and PCC are the correct predictions for each SS conformation and N is the sum of all the parameters in the matrix. Moreover, from the binary confusion matrix, for each secondary structure, the main binary scoring indexes have been computed :

- Sensitivity, a measure of how many of the real examples are correctly predicted.

  $$SEN = \frac{TP}{TP + FN}$$

- Positive predictive value, a measure of how many of the positive predictions are correct.

  $$PPV = \frac{TP}{TP + FP}$$

- Matthews correlation coefficient, it assumes 0 value for random predictions, 1 for perfect predictions and -1 value for completely wrong predictions.

  $$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP+FP) \times (TP+FN) \times (TN+FP) \times (TN+FN)}}$$

- The Segment OVerlap index (SOV), which evaluates the percent average overlap between predicted and observed segments in the three SS conformations. The advantage of this index is that it considers the size of continuous overlapping segments and assigns extra allowance to longer continuous overlapping segments (instead of only judging from the percentage of overlapping individual positions as Q3 score does)[Liu and Wang, 2018].

  $$\text{SOV(S)} = 100 \times \frac{1}{N_S} \sum_{(S_O,S_P)|S_O \cap S_P \neq 0} \left[\frac{minov(S_O,S_P) + \sigma(S_O,S_P)}{maxov(S_O,S_P)} \times len(S_O)\right]$$

  Where:
  N_s is the number of residues in conformation S;
  $minov(S_O, S_P) = S_O \cap S_P$ is the length of the overlap of S_O and S_P;
  $(S_O, S_P)|S_O \cap S_P \neq 0$ is the set of observed and predicted segments which overlaps by at least one residue;
  $maxov(S_O, S_P) = len(S_O) + len(S_P) - S_O \cap S_P$ is the length of the total extent of S_O and S_P;

  $$\sigma(S_O, S_P) = min \begin{cases} maxov(S_O, S_P) - minov(S_O, S_P) \\ minov(S_O, S_P) \\ len(S_O)/2 \\ len(S_P)/2 \end{cases}$$

  Is a correction introduced in the last version of the index to cope with uncertainty in experimental segment boundary assignment.

3.2 Cross-validation on training set

Cross-validation is a re-sampling procedure applied to evaluate machine learning models on a limited data sample. The goal of cross-validation is to test the model's ability to predict new data that have not been used during the training, in order to avoid overfitting. It gives an idea on whether the produced model will be general enough to correctly predict an independent dataset (our blind test set, that has never been seen during the training procedure). This approach consist in randomly dividing the set of observations (our training set) into k groups of approximately equal size. For our purpose, the number of groups in which the training set has been split into was 5 so that each train/test group of data samples is large enough to be statistically representative of the broader dataset.

SVM Greed search	$\gamma = 0,5$ C=2	$\gamma = 2$ C=2	$\gamma = 0,5$ C=4	$\gamma = 2$ C=4
PERFORMANCE H	SEN= 0.681± 0.010 PPV= 0.845 ± 0.004 MCC=0.648 ± 0.007 SOV= 79.236 ± 0.834	SEN= 0.122 ± 0.004 PPV= 0.874 ± 0.009 MCC= 0.247 ± 0.006 SOV= 15.881 ± 0.405	SEN= 0.673 ± 0.010 PPV= 0.842 ± 0.004 MCC= 0.641 ± 0.008 SOV= 79.075 ± 0.8	SEN= 0.12 ± 0.005 PPV= 0.873 ± 0.009 MCC= 0.245 ± 0.007 SOV= 15.672 ± 0.413
PERFORMANCE C	SEN= 0.878 ± 0.002 PPV= 0.618 ± 0.003 MCC= 0.484 ± 0.003 SOV=68.490 ± 1.177	SEN= 0.984 ± 0.000 PPV= 0.441 ± 0.002 MCC= 0.151 ± 0.004 SOV= 11.028 ± 0.611	SEN= 0.877 ± 0.002 PPV= 0.611 ± 0.003 MCC= 0.474 ± 0.004 SOV= 68.886 ± 1.162	SEN= 0.985 ± 0.001 PPV= 0.441 ± 0.003 MCC= 0.15 ± 0.005 SOV= 10.952 ± 0.597
PERFORMANCE E	SEN= 0.397 ± 0.007 PPV=0.791 ± 0.0072 MCC=0.486 ± 0.0046 SOV= 48.735 ± 0.106	SEN= 0.019 ± 0.003 PPV= 0.822 ± 0.03 MCC= 0.103 ± 0.009 SOV= 2.478 ± 0.577	SEN= 0.387 ± 0.007 PPV=0.788 ± 0.007 MCC= 0.477 ± 0.004 SOV= 47.915 ± 0.07	SEN= 0.019 ± 0.003 PPV= 0.814 ± 0.028 MCC= 0.101 ± 0.009 SOV= 2.436 ± 0.584
Q3	0.702 ± 0.002	0.464 ± 0.003	0.696 ± 0.003	0.464 ± 0.003

Table 1. Performance of SVM: this table has been generated with the average values of the training set in cross validation. As we can see the best prediction is the one obtained with the hyperparapetres $\gamma 0,5 C2$

Cross validation performance	GOR	SVM
PERFORMANCE H	SEN= 0.787±0.006 PPV= 0.636±0.008 MCC=0.519 ±0.006 SOV= 76.528 ±0.834	SEN= 0.681±0.010 PPV= 0.845 ±0.004 MCC=0.648 ±0.007 SOV= 79.236 ±0.834
PERFORMANCE C	SEN= 0.432 ±0.011 PPV= 0.804 ±0.005 MCC= 0.418 ±0.004 SOV=54.94 ±0.365	SEN= 0.878 ±0.002 PPV= 0.618 ±0.003 MCC= 0.484 ±0.003 SOV=68.490 ±1.177
PERFORMANCE E	SEN= 0.709 ±0.016 PPV=0.473±0.011 MCC=0.428 ±0.005 SOV= 69.622 ±0.106	SEN= 0.397 ±0.007 PPV=0.791 ±0.0072 MCC=0.486 ±0.0046 SOV= 48.735 ±0.106
Q3	0.619 ±0.003	0.702 ±0.002

Table 2. Shows the averages performance values in cross validation and compares the results for the two methods.

Blind set performance	GOR	SVM
PERFORMANCE H	SEN= 0.728 PPV= 0.653 MCC=0.464 SOV= 70.478	SEN= 0.601 PPV= 0.879 MCC= 0.600 SOV= 71.440
PERFORMANCE C	SEN= 0.415 PPV= 0.737 MCC= 0.390 SOV=52.370	SEN= 0.890 PPV= 0.543 MCC= 0.446 SOV= 64.017
PERFORMANCE E	SEN= 0.707 PPV=0.470 MCC=0.415 SOV= 67.531	SEN= 0.429 PPV= 0.821 MCC= 0.519 SOV= 52.174
Q3	0.607	0.668

Table 3. Shows the performances of both the methods on the blind set.

For each group, the first fold was treated as a validation set, and the model was trained on the remaining 4 folds (k-1 folds). We evaluated the performance of GOR and SVM models on the training set, using a 5-fold crossvalidation (results are reported in supplementary materials). In Table 2 are reported the resulting performance of GOR in cross validation. All the reported indexes represent the computed averages accompanied by the cor-respective Standard deviation Error . defined as:

$$SE = \frac{\sigma}{\sqrt{n}} \tag{20}$$

where σ is the standard deviation and n is the number of samples, in our case is 5 since we have 5 different cross validation folds.
To evaluate of the SVM performance for each of the 4 set of hyperparameters we tested in cross validation,we summarize all the measures in supplementary materials.

In Table 1 we reported the averages values for MCC and SOV indexes computed for each of the 4 couples of hyperparapeters. The results show a better overall performance obtained in cross validation with $\gamma = 0.5$ and $C = 2$ so this moved our choice to use these values for the SVM prediction on the blind set. As we can notice from Table 2, considering the SOV index, for the GOR method coils is the structure that is predicted the worst. The SVM method instead performs the worst results with strands.

3.3 Blind test set prediction

We used both the GOR and SVM methods to predict the secondary structure of sequences belonging to the Blind test (results reported in Table 3. In general, these results are consistent with the performances in cross-validation. The computed performances for the prediction of the blind test set showed:

- A better accuracy of SVM method (Q3= 0.668) with respect to the one of GOR (Q3= 0.607)
- The MCC computed for all the three classes proved to be higher for the SVM prediction than for the GOR one (as resulting by Table 3.
- From the comparison of the SOV index computed for each class with both the methods arose interesting properties: The overall average SOV index for the SMV method results to be lower than the one computed for GOR. Specifically helix resulted to be predicted quite well from both the methods, but it appears that SVM fails in the prediction of strands and instead coil is worst predicted structure for GOR.

4 Discussion

The necessity of methods for the predictions of the secondary structures finds its cause in the significant improvements of structure determination techniques. They lead to nothing more than an increasing gap between

the number of protein structures in public databases, and the number of known protein sequences. Moreover, secondary structure becomes important while performing functional annotation in silico by homology search; as conserved structure are much more informative than sequences. From the analysis of the performance carried out for both the methods, appears that SVM performs a better overall prediction, more accurate and more precise than the one of GOR. Despite this, the SVM method showed an high sensitivity in predicting helix and coil conformations but the strand values were estimated quite low. The contrasting results for SOV indexes regarding the prediction of strands by SVM can be easily explained looking at the biological features for the formation of this structure: The influence of long-range residue interactions on defining secondary structure can be a strong limitation for secondary structure prediction. In fact the method we tested assign a secondary structure to a window of a local segment and thus do not explicitly consider long-range interactions of amino acids. Thus, it is not surprising that βstrand prediction lead to some problem, because its formation is mainly due to long-range interactions [Kihara, 2005].

References

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990). Basic local alignment search tool. *Journal of molecular biology*, **215**(3), 403–410.

Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. (1997). Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic acids research*, **25**(17), 3389–3402.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, **2**, 27:1–27:27. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

Chou, P. Y. and Fasman, G. D. (1974). Conformational parameters for amino acids in helical, β-sheet, and random coil regions calculated from proteins. *Biochemistry*, **13**(2), 211–222.

Dondoshansky, I. and Wolf, Y. (2002). Blastclust (ncbi software development toolkit). *NCBI, Bethesda, Md*, **14**.

Drozdetskiy, A., Cole, C., Procter, J., and Barton, G. J. (2015). Jpred4: a protein secondary structure prediction server. *Nucleic acids research*, **43**(W1), W389–W394.

Engelking, L. R. (2014). *Textbook of veterinary physiological chemistry*. Academic Press.

Garnier, J., Osguthorpe, D. J., and Robson, B. (1978). Analysis of the accuracy and implications of simple methods for predicting the secondary structure of globular proteins. *Journal of molecular biology*, **120**(1), 97–120.

Gribskov, M., McLachlan, A. D., and Eisenberg, D. (1987). Profile analysis: detection of distantly related proteins. *Proceedings of the National Academy of Sciences*, **84**(13), 4355–4358.

Jones, S. and Thornton, J. M. (1996). Principles of protein-protein interactions. *Proceedings of the National Academy of Sciences*, **93**(1), 13–20.

Kabsch, W. and Sander, C. (1983). Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers: Original Research on Biomolecules*, **22**(12), 2577–2637.

Kihara, D. (2005). The effect of long-range interactions on the secondary structure formation of proteins. *Protein Science*, **14**(8), 1955–1963.

Lim, V. (1974). Algorithms for prediction of α-helical and β-structural regions in globular proteins. *Journal of Molecular Biology*, **88**(4), 873–894.

Liu, T. and Wang, Z. (2018). Sov_refine: A further refined definition of segment overlap score and its significance for protein structure similarity. *Source Code for Biology and Medicine*, **13**(1), 1.

RCSPDB (2019). worldwide PDB. www.wwpdb.org .

Rost, B., Sander, C., and Schneider, R. (1994). Redefining the goals of protein secondary structure prediction. *Journal of molecular biology*, **235**(1), 13–26.

UniProt (2019). Uniprotkb/swissprot. http://www.uniprot.org.