Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: 14/12/2020
Scientific article

Structural bioinformatics

GOR method vs Support Vector Machines for predicting protein secondary structure from amino-acid sequence

Rosaria Tornisiello [1,*]

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

[*] To whom correspondence should be addressed.

Abstract

Motivation: While the advent of large-scale sequencing techniques led to an exponential increase of protein sequence data, the experimental determination of protein structure is still in development. Being able to predict protein structure is the key issue for protein functional annotation. Therefore, several structure prediction methods have been developed over the last 70 years. Here we compare the performances of GOR method (Garnier-Osguthorpe-Robson) and Support Vector Machines (SVM) which are respectively based on Bayesian statistical analysis and machine learning.
Results: The results show how the machine learning based method performs slightly better than the GOR method, giving a more accurate prediction of the test set.
Contact: rosaria.tornisiello@studio.unibo.it
Supplementary information: Supplementary data are available at *https://github.com/RosariaTornisiello/lb2-2020-project-Tornisiello*

1 Introduction

Proteins are biopolymers composed of one or more aminoacidic chain. These macromolecules perform several functions such as catalysis, molecular recognition, structural support, intracellular transport and DNA replication, thanks also to their capacity to fold in multiple three-dimensional shapes. This suggests how knowing only the protein sequence is not sufficient to understand protein function. With the advent of next-generation sequencing techniques over 200 million protein sequences have been identified. Instead, given the high cost and the degree of difficulty of experimental protein structure determination, only about 170000 structures have been deposited (Yang *et al.*, 2018). Although prediction of tertiary structure is one of the ultimate goals of protein science, the prediction of secondary structure from sequence is still a more feasible intermediate step in this direction. Moreover, knowledge about the secondary structure allows to adopt a divide and conquer approach to solve the full three-dimensional structure. This method is based on the prediction of simplified aspects of the structure, namely the key structural elements of the protein and the location of these elements not in the three-dimensional space but along the protein amino acid sequence. Consequently, the accuracy of protein secondary structure prediction directly impacts the accuracy of the overall protein structure determination (Yang *et al.*, 2018). Moreover, secondary structure prediction plays a central role in the classification of unstructured and intrinsically disordered proteins. Secondary structure prediction method are classified in three categories:

- First-generation methods: secondary structure was predicted based on the sequence according to the propensity of each amino acid residue to a specific secondary structure element. The representative method of this category is the Chou–Fasman method (Chou and Fasman, 1974).
- Second-generation methods: used a sliding window of 3-51 neighbouring residues and statistical information. This class is represented by the GOR method that we will describe later (Garnier *et al.*, 1978).
- Third-generation: predicted secondary structure by using evolutionary information derived from multiple alignment of homologs sequences. The most popular predictor available of this generation is PSIPRED (Jones, 1999) which uses a two-stage neural network and is based on the position specific scoring matrices generated by PSI-BLAST. Other methods belonging to this group are: SSPro2 (Magnan and Baldi, 2014), JPred4 (Drozdetskiy *et al.*, 2015) and PHDsec/PROFsec (Rost and Sander, 1993).

Accuracy improved with advancing generation firstly thanks to the inclusion of neighbouring residues within the window and secondly thanks to the usage of evolutionary information. In this project we compare second-generation and third-generation methods: GOR method and SVM. Both methods are trained on the JPred4 dataset (Drozdetskiy *et al.*, 2015), the trained model are validated by means of a 5-fold cross-validation (CV) and finally tested on a blind test set. The performances are evaluated according to Matthews correlation coefficient, precision, recall and accuracy. Our results show that SVM is able to predict the secondary structure conformation of our test set with an higher accuracy than the GOR method by 4%, performing particularly better in the prediction of the Helix conformation.

2 Materials and Methods

2.1 Training dataset

2.1.1 General description

The GOR model and the SVM were trained both on the same training dataset used for the JPred4 prediction method. This dataset was generated by selecting the representatives from SCOP (Fox Naomi and Brenner Steven, 2014) super-family level. Given that the SCOP classification is structure-based, this approach lessens the probability of including trivially detectable sequence similarities in the training dataset. The production of the JPred4 dataset started with 1987 representative sequences for each super-family in SCOPe v.2.04. Sequences are filtered and removed thereafter, according to the following criteria:

- structure resolution >2.5 Angstrom;
- sequences with length <30 residues since they are unlikely to represent globular protein domains;
- sequences with length >800 residues due to the time required for building the sequence profiles;
- multi-chain or fragment domains;
- sequences with missing DSSP information for >9 consecutive residues;
- inconsistencies between PDB, DSSP and other file definitions.

The described filtering process left a total of 1507 domain sequences. Afterwards, the obtained dataset was separated into two parts: a training set with 1357 sequences and a blind-test set composed by 150 sequences. In our work the blind-test set was generated in a separate pipeline.

2.1.2 Statistical analysis of JPred4 dataset

In order to inspect the training set and check if it was a good representative of the protein space, we performed basic statistical analysis by means of a python pipeline developed in house, and compared the results with the UniprotKB/Swiss-Prot statistics as of October 2020. We started analysing the taxonomic classification of the training set, shown as a pie plot in figure 1. Comparing our Super-kingdom distribution with the one shown in the

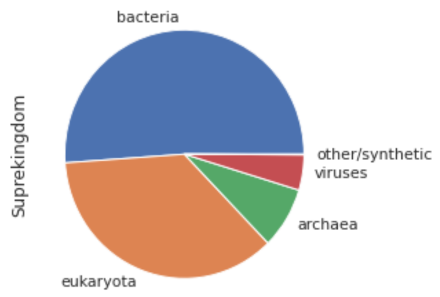

Fig. 1. Pie plot that shows the taxonomic classification of the JPred4 dataset, obtained using the pandas package.

SwissProt statistics, it is clear that the proportions are maintained. The only noticeable difference is the presence of synthetic proteins that were not removed from the JPred4 dataset. The amino-acidic and the secondary structure composition of the training dataset are illustrated in the barplot in figure 2 and in the pieplot in figure 3, respectively. Both the distributions are comparable to the SwissProt ones, however, figure 3 shows an evident bias towards the helix composition due to the approximated grouping of secondary structure conformations described in section 2.2.2.

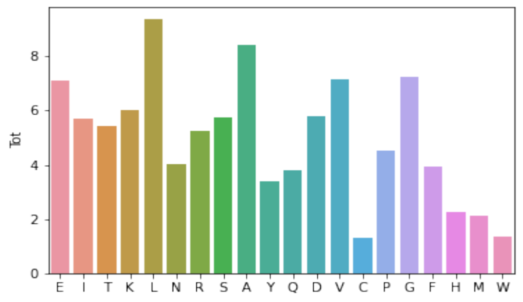

Fig. 2. Barplot that shows the amino-acidic composition of the JPred4 dataset, obtained using the seaborn package.

The barplot in figure 4 shows the propensity of each amino-residue type towards the three different secondary structure conformations. In particular the amino-acid types are ordered from left to right according to the physio-chemical properties: the first group is composed by apolar residues, followed by the aromatic group, the polar group and the charged group. This configuration allows to highlight the correlation between physio-chemical properties and secondary structure propensity of each residue type. The amino acids that tend to form α-elix are: Ala, Leu, Glu, Gln, Arg, Met, Lys; instead, Val, Ile, Tyr, Phe, Thr and and Trp are prone to be in β-strands. Cys and His tend to be unstructured residues and all the remaining ones are likely to be found in the coil conformation. Alpha-

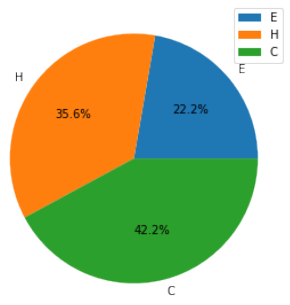

Fig. 3. Barplot that shows the secondary structure composition of the JPred4 dataset, obtained using the pandas package.

helix admirers do not have polar heteroatoms on C_β and C_γ atoms, nor branching or aromatic group on C_β atom. Amino acids that have aromatic groups or branching on C_β atom are strand admirers. Turn and bend admirers have polar heteroatom on C_β or C_γ atoms or do not have C_β atom at all (Malkov *et al.*, 2005). These correlations seem to be approximately maintained in our dataset, as highlighted by figure 4 (considering turns and bends belonging to the coil class). As it is evident from the figure 5, the training set covers all the SCOP classes. Moreover, we generated a probability density distribution of the protein sequences length of this dataset that is available in the supplementary materials. Finally we can confirm the goodness of the JPred4 dataset, given its high similarity with the SwissProt database.

2.2 Blind-test dataset

2.2.1 Selection of 150 entries

In order to evaluate the performance of both models, a separate blind-test set was generated, which is composed by samples totally independent

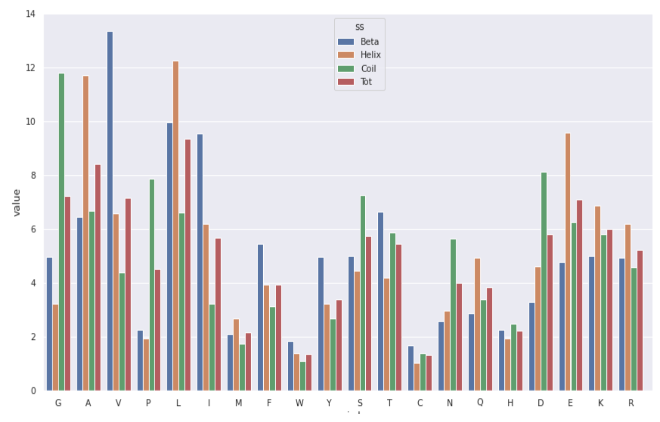

Fig. 4. Barplot that shows the secondary structure distribution for each amino-acid type of the JPred4 dataset, obtained using the seaborn package.

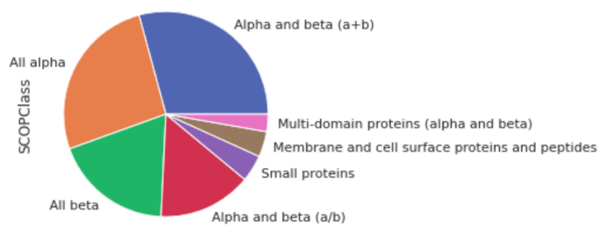

Fig. 5. Pie plot that shows the relative abundance of each SCOP class in the training dataset, obtained using the pandas package.

from the training set so as to avoid prediction biases. First of all, we carried out an advanced search on RCSB PDB (Berman *et al.*, 2000) setting the following filters:

- release date >= 01/01/2015;
- experimental method: X-ray diffraction;
- refinement resolution <= 2.5 Angstrom;
- polymer entity sequence length between 50 and 300 residues;
- polymer entity type: protein.

Applying these filters we ended up with 24714 PDB entries. Then we downloaded the PDB ids list and retrieved the correspondent fasta sequences. Implementing a python pipeline, we checked again the length of the single fasta chains and their composition, removing the ones with less then 50 and more that 300 residues length and the ones containing unidentified residues ("X" characters). In order to avoid over-representation of certain SCOP classes, we lowered the internal redundancy below 30% by clustering the remaining chains using MMseqs2 (Many-against-Many sequence searching) which is a software suite to search and cluster huge protein and nucleotide sequence sets (Steinegger and Söding, 2017). The chosen clustering criteria were: minimum sequence identity of 30%, length coverage threshold equals to 0.5 and an e-value threshold of 0.001. Following these steps we obtained 4760 representative protein entities. The last filtering step was the removal of the external redundancy through BLASTp (Altschul *et al.*, 1990). Firstly, the JPred4 dataset was formatted as database with the *makeblastdb* tool, then all the cluster representatives were compared to it setting an e-value threshold of 0.01, finally all entries with a sequence identity equals or higher than 30% were excluded. In order to build the final blind-test set, 150 random sequences were selected from the remaining ones and their matching PDB files were downloaded.

2.2.2 Retrieving the secondary structure

By means of an in-house developed bash pipeline, we further skimmed the structure files obtained in the previous steps, selecting only the protein chains of interest. The following procedure was the usage of the DSSP program (Kabsch and Sander, 1983) to derive the secondary structure of the selected chains. The DSSP program defines secondary structure, geometrical features and solvent exposure of proteins, given atomic coordinates in Protein Data Bank format (PDB) or macro-molecular Crystallographic Information File format (mmCIF). DSSP is able to assign the following 8 different secondary structure conformations: α-helix (H), β-bridge residue (B), extended strand (E), 3/10-helix (G), 5-helix (I), H-bonded turn (T), bend (S) or empty. In order to simplify the classification, in our work we grouped them as:

- H, G and I classified as H;
- B and E classified as E;
- T, S and empty classified as C or "-".

Once obtained the DSSP output files, both the secondary structure sequence and the primary sequence were extracted in FASTA format by means of an in-house developed python script.

2.2.3 Statistical analysis of the blind-test set

As for the training dataset, some basic statistical analysis have been performed to inspect the sequence length, the amino-acidic composition and the secondary structure conformation distribution of the test set. The resulting plots are available in the supplementary materials. It is possible to confirm that our test set has been generated in compliance with the JPred4 dataset and with the protein space properties.

2.3 Sequence profiles generation

As previously said, new generation methods for predicting secondary structure avail of evolutionary information to improve the prediction performance. In particular they are based on the use of sequence profiles obtained from multiple sequence alignments (MSA). Since sequence profiles summarize the MSA from which they are derived, gathering information about protein families including also distantly related homologs, they enable the predictor to reach a much higher accuracy than the simple prediction of the first-generation methods. In our work, in order to obtain the sequence profiles of both the training and test set, we searched all the sequences belonging to them in a large sequence database obtained indexing the UniprotKB/SwissProt database (Consortium, 2019) using the *makeblastdb* tool. Then the PSI-BLAST (Position-Specific Iterative BLAST, (Altschul *et al.*, 1997)) algorithm was used to search for homologs sequences in the obtained database. The procedure PSI-BLAST uses can be summarized in five steps:

1. PSI-BLAST takes as an input a single protein sequence and compares it to a protein database, using the gapped BLAST program.
2. The program constructs a multiple alignment, and then a profile, from any significant local alignments found.
3. The program compares the profile to the protein database, again seeking local alignments. After a few minor modifications, the BLAST algorithm can be used for this directly.

4. PSI-BLAST estimates the statistical significance of the local alignments found.
5. Finally, PSI-BLAST iterates, by returning to step (2), an arbitrary number of times or until convergence.

PSI-BLAST uncovers many protein relationships missed by single-pass database-search methods. In our case PSI-BLAST was executed setting an e-value threshold equal to 0.01, maximum number of iterations equal to 3 and number of descriptions equal to 10000. This program returns as output for each sequence given as input, the Position-Specific Substitution Matrix (PSSM) computed during the search and the file containing the pairwise alignments. In order to extract the sequence profiles, the PSSM files are then parsed through a python script developed in house. Finally, for both the training and the blind-test set the sequence profiles were parsed removing all the 0 matrices arose due to PSI-BLAST not finding any similarity in the database. At the end of the described procedure, we obtained 1204 training profiles and 117 test profiles.

2.4 The GOR method

2.4.1 General overview

The GOR program is one of the first major methods proposed for protein secondary structure prediction from sequence. The original article (GOR I) was published by Garnier, Osguthorpe, and Robson in 1978, however, the method has been continuously improved and modified for 30 years with the last GOR V version published in 2002. The GOR algorithm was modified to include the triplet statistics within the window. The previous versions of the program used only single residue statistics (GOR I–II) or the combination of the single residue and pair residue statistics within the window (GOR III–IV). The most important improvement was the inclusion of multiple sequence alignments from PSI-BLAST program, which allows to take into account evolutionary information for the secondary structure prediction (Kloczkowski *et al.*, 2002). The GOR algorithm is based on information theory combined with Bayesian statistics. One of the basic mathematical tools of information theory is the information function I(S,R):

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

For the problem of protein secondary structure prediction, the information function is defined as the logarithm of the ratio of the conditional probability P(S|R) of observing conformation S, (where S is one of the three states: helix (H), extended (E), or coil (C)) for residue R (where R is one of the 20 possible amino acids) and the probability P(S) of the occurrence of conformation S. The conformational state of a given residue in the sequence depends not only on the type of the amino acid R but also on the neighboring residues along the chain within the sliding window. GOR IV used a window of 17 residues, that is, for a given residue, eight nearest neighboring residues on each side were analyzed. According to information theory, the information function defined over the window of residues centered at a given residue position R_0, considering d equal to 8, is expressed as:

$$I(S;R_{-d},\ldots,R_d) = \log \frac{P(S|R_{-d},\ldots,R_d)}{P(S)} = \log \frac{P(S,R_{-d},\ldots,R_d)}{P(R_{-d},\ldots,R_d)P(S)} \qquad (2)$$

Computing all the joint probabilities of S and all the possible residues in all the positions of the window is computationally expensive since we would have an exponential number of possible configurations. Moreover, we would need a very large sequence database to estimate reliable distributions. The solution of this problem is to assume statistical independence of all the 17 residues within the window. Given this statement, the previous formula can be expressed as:

$$I(S;R_{-d},\ldots,R_d) \approx \sum_{k=-d}^{d} I(S;R_k) \qquad (3)$$

The window-based information function is equal to the sum of the single residue functions. The parameters needed to compute the equation (3) are:

- $P(R_k,S)$: probability of observing a residue of type R at position k in the window and the central residue R_0 in conformation S:
- $P(R_k)$: probability of observing a residue of type R at position k in the window;
- $P(S)$: probability of observing conformation S.

The GOR method training phase consists in the computation of a matrix containing the information function for each position in the sliding window, considering all kinds of secondary structure conformation and residue. The information matrix is then used to predict the secondary structure of a test set of protein sequences, given their profiles. During the prediction phase, each residue position of all the query sequences is analyzed, associating it with the conformation S characterized by the highest value in the information matrix, that is:

$$S = \arg\max_S \sum_{k=-d}^{d} I(S;R_k) \qquad (4)$$

2.4.2 Implementation

In this project we implemented the GOR model by means of two python pipelines developed in-house. The first python script is dedicated to the training of the model, that is the generation of the information function. It takes as input the training set UniProt IDs list, the training set sequence profiles (in which we inserted the secondary structure sequence defined by DSSP), and gives as output the information matrix. The latter is given as input to the prediction python script, along with the test set UniProt IDs list and their sequence profiles. The final outputs of our GOR method implementation are the predicted secondary structure sequences of all the test set entries, stored in FASTA files.

2.5 Support Vector Machines

2.5.1 General description and geometrical interpretation

A Support Vector Machine (SVM) is a supervised machine learning approach that is able to perform both classification and regression analysis. For the purpose of this project, we will focus on the Support Vector Classifiers (SVC). The goal of a SVC is to separate two classes of data using a function which is induced from the training examples. The final classifier must be able to generalize unseen data.

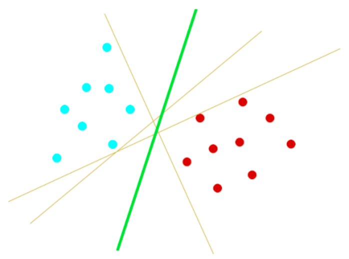

Fig. 6. Optimal separating hyperplane (Gunn et al., 1998).

Considering the example in figure 6, we can see that it is possible to draw several separating linear classifiers, however, only one of them maximizes the distance between itself and the nearest points of each class: this classifier is defined as the optimal separating hyperplane. The latter is defined as the set of points $\vec{x}$ characterized by the same projection on a given vector $\vec{w}$:

$$\vec{w}\vec{x} + b = 0 \tag{5}$$

A separating hyperplane in canonical form must satisfy the following constraints:

$$y^i[\langle w, x^i \rangle + b] \geq 1, i = 1, \ldots, l. \tag{6}$$

The distance $d(w, b; x)$ of a point x from the hyperplane (w, b) is:

$$d(w, b; x) = \frac{|\langle w, x^i \rangle + b|}{\|w\|} \tag{7}$$

The optimal separating hyperplane will be the linear classifier that maximizes the margin ρ, that is the one that minimizes $\|w\|$. It is possible to demonstrate that ρ is equal to $\frac{2}{\|w\|}$; hence the hyperplane that optimally separates the data is the one that minimises:

$$\Phi(w) = \frac{1}{2}\|w\|^2 \tag{8}$$

The solution to the optimisation problem of Equation 8 under the constraints of Equation 6 is given by the saddle point of the Lagrangian:

$$L(\vec{w}, b, \alpha_i) = \frac{1}{2}\|w\|^2 + \sum_i \alpha_i[1 - y^i(\langle w, x^i \rangle + b)] \tag{9}$$

The minimization of the Lagrangian (concave function) can be converted in the maximization of the Dual Lagrangian (convex function):

$$\begin{aligned} \bar{L}(\alpha_i) &= -\frac{1}{2}\sum_i\sum_j \alpha_i\alpha_j y^i y^j x^i x^j + \sum_i \alpha_i \\ \alpha_i&[1 - y^j(\langle w, x^i \rangle + b)] = 0 \,\forall\, i \\ \alpha_i &\geq 0 \,\forall\, i \end{aligned} \tag{10}$$

SVC as described above is defined as *hard margins SVC* since no margin violation is allowed. However, frequently, a *soft margins SVC* is preferred because it produces a real valued output between -1 and 1 when the classifier is queried within the margin, where no training data resides (Gunn *et al.*, 1998). In this case we introduce the *slack variable* ξ_i to allow some misclassification:

$$\begin{aligned} minimize \quad & \Phi(w, \xi) = \frac{1}{2}\|w\|^2 + C\sum_i \xi_i \\ constrained\ by \quad & y^i[\langle w, x^i \rangle + b] \geq 1 - \xi_i \quad \xi_i \geq 0, \forall i \end{aligned} \tag{11}$$

Where C is an hyperparameter that represents a trade-off between the minimization of ξ and the maximization of ρ. When data are not linearly separable, we can apply the *kernel trick*: the idea is mapping the non-linearly separable dataset into a higher dimensional space where we can find an hyperplane that can separate the samples. As it is possible to observe in the Equation 10, the vector $\vec{x}$ appears only in the scalar product, consequently we can deduce that the optimization problem depends only on it. Hence, we can apply the kernel trick without explicitly computing the coordinates of the transformed data points. If we use a mapping function that maps our data into a higher dimensional space, then, the maximization will depend on the scalar products $\phi(\vec{x})\phi(\vec{y})$ of the mapping function for different samples. This scalar product is also defined as kernel function $K(\vec{x}, \vec{y})$. The SVC implemented for this project uses the radial basis function (rbf) kernel that is formulated as follows:

$$K(\vec{x}, \vec{y}) = e^{(-\gamma\|\vec{x}-\vec{y}\|^2)} \tag{12}$$

Where γ is an hyperparameter.

2.5.2 Application of SVM for secondary structure prediction

The problem of secondary structure prediction has been addressed by means of several machine learning models including multi-layer perceptrons, recurrent neural networks and SVMs which have proven to be promising classifiers. Indeed, SVMs are successful in controlling the classifier's potential for overfitting ensuring the maximization of ρ (Ward *et al.*, 2003). Other convenient aspects of SVMs are the use of few hyperparameters and the interchangeable use of kernel functions. Nevertheless, SVMs are not designed to perform multi-class classification analysis as the task at hand, but only binary ones. This issue is faced applying either the One-vs-Rest (OvR) or the One-vs-One (OvO) strategy: in the first case the multi-class classification is split into one binary classification problem per class, in the second case the multi-class classification is split into one binary classification problem per each pair of classes. Hence, the application of SVM for multiclass prediction using the OvR strategy consists in training N binary SCV, where N is the number of classes, or (N(N-1))/2 binary SVC adopting the OvO option. In our work, we chose to train the SVC using the OvR approach.

2.5.3 Grid search

Grid search is a process of hyperparameter optimization, i.e. the exploration of all the hyperparameter combinations to find the ones that yields the best machine learning predictor. In our work we performed a grid search analysis testing a C parameter equal to 2 or 4 and a γ equal to 0.5 and 2. In principle, the grid search should explore also the kernel functions space, however, in our project we chose to maintain the default kernel function of the sklearn SVC (rbf) due to time constraints and limited computational power.

2.5.4 Implementation

For the purpose of this project, training and testing of the SVC were both implemented using the scikit-learn (sklearn) python module, version 0.24 (Pedregosa *et al.*, 2011), that is based on the LIBSVM library (Chang and Lin, 2011). The sklearn implementation allows to tune both the hyperparameters C and γ and to set the desired kernel function. As regards the training phase, we started producing our SVC inputs i.e. the matrix of features and the array containing the classes of each sample of the training dataset. Each sequence profile, obtained as described in section 2.3, was parsed by means of a sliding window; each window was flattened and finally, all the windows were concatenated to generate the final matrix of features. Moreover, the secondary structure conformation defined by DSSP of each residue position was stored in an array, modifying the class attributes as follows: class 1 for Helix, class 2 for Strand and class 3 for Coil. Thereafter, we performed the fitting of our SVC. The prediction phase was then carried out, handling the test dataset with the very same procedure adopted for the training dataset.

2.6 Evaluation procedure and scoring measure

2.6.1 Scoring indexes

In order to evaluate and compare the performances of both predictors we computed the following scoring indexes:

- Q3 score, the multiclass accuracy which gives the percentage of amino acids for which the secondary structure conformation was correctly predicted (Spencer *et al.*, 2014). Considering two classes i and j with $1 < i, j < k$, we can compute the Q3 score as:

$$Q_3 = \frac{p_{11} + p_{22} + \cdots + p_{kk}}{N} \tag{13}$$

where $p_{i,j}$ is an object of class i predicted in class j and N is the number of classes.

- The Matthews correlation coefficient (MCC) that is a more reliable statistical rate which accounts for all of the four categories true positives (FP), false negatives (FN), true negatives (TN), and false positives (FP). When the classifier is perfect (FP = FN = 0) the value of MCC is 1, indicating perfect positive correlation. Conversely, when the classifier always misclassifies (TP = TN = 0), we get a value of -1, representing perfect negative correlation. In contrast with the accuracy, this rate is not affected by class imbalance. It is computed as:

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{14}$$

- Precision, also called positive predicted value, is the fraction of true positive over all the positive predictions. It is equal to:

$$\text{Precision} = \frac{TP}{TP+FP} \tag{15}$$

- Recall, also called sensitivity, that is the proportion of actual positives correctly identified:

$$\text{Recall} = \frac{TP}{TP+FN} \tag{16}$$

The cross validation results are reported along with the mean and standard error that is computed as:

$$\text{Standard error} = \frac{\sigma}{\sqrt{n}} \tag{17}$$

Where σ is the standard deviation and n is the number of samples, that in our case is equal to 5.

2.6.2 Cross-validation

CV is a machine learning technique used to assess how accurately the model will predict a previously unseen dataset. The CV methods can be classified in two sub-categories:

- Exhaustive CV: learn and test the model on all possible ways to divide the original sample into a training and a validation set;
- Non-exhaustive CV: do not compute all the possible ways of splitting the original sample.

In this project we performed a non-exhaustive, k-fold CV, by which the entire training set is partitioned in k equal sized subsets. Each subset is used as validation set to test the model by rotation, while the other $k - 1$ subsets serve as training dataset. At the end of the process, the k results are averaged to produce a single evaluation.

3 Results

3.1 Cross-validation results

The 5-fold CV of both methods has been evaluated by means of the MCC computation. Table 1 lists the MCC scores computed for each fold of the GOR CV. While Table 2 stores the MCC scores for each fold of the 4 different SVC tested by means of grid search. For both methods, the mean is reported along with the standard error.

As for the GOR method, the CV yielded a mean MCC of 0.461 and a standard error of 0.002. The 5 scores show no significant difference and a very low standard error meaning that the performance of the model for each fold is approximately equivalent. The slight difference between the MCC values is not surprising, given the subtle discrepancy in amino-acid composition and secondary structure distribution between the 5 subsets. The SVM CV was performed for each combination of the C and γ parameters. The classifier that yielded the maximum MCC value was the one characterized by $C = 2$ and $\gamma = 0.5$, with a mean MCC equal to 0.548 and a very low standard error of 0.003.

	fold 0	fold 1	fold 2	fold 3	fold 4	Mean
MCC	0.460	0.466	0.468	0.454	0.457	0.461 ± 0.002

Table 1. 5-fold cross-validation MCC values, mean and standard error of GOR model.

Fold	$C = 2$, $\gamma = 0.5$	$C = 2$, $\gamma = 2$	$C = 4$, $\gamma = 0.5$	$C = 4$, $\gamma = 2$
0	0.549	0.168	0.538	0.166
1	0.552	0.185	0.544	0.181
2	0.560	0.184	0.549	0.181
3	0.538	0.168	0.529	0.166
4	0.540	0.153	0.530	0.150
Mean	0.548±0.003	0.171±0.004	0.538±0.003	0.169±0.004

Table 2. 5-fold cross-validation MCC values, mean and standard errors for each SVC tested by means of grid search.

3.2 Blind test results

Table 3 lists all the scoring indexes values for each secondary structure conformation of both models, obtained by the prediction phase of the blind test set. The MCC values are higher for the H conformation in both cases. However, it is possible to notice that the GOR model precision is higher for the C conformation, while the SVM is more precise in predicting the H conformation. Instead, concerning the recall, we have the inverse situation. The SVM prediction shows an overall three classes accuracy, Q3, higher than the GOR model.

SCORE	SS	GOR	SVM
MCC	H	0.515	0.637
	E	0.466	0.517
	C	0.426	0.473
Precision	H	0.654	0.857
	E	0.551	0.813
	C	0.744	0.566
Recall	H	0.787	0.669
	E	0.683	0.444
	C	0.464	0.872
Q3	total	0.643	0.687

Table 3. Blind-test set scores for both the GOR and the SVM models, reported for each secondary structure conformation.

3.3 Discussion of results

Firstly, we can observe that there is not a substantial difference in the MCC values between CV and test phase, meaning that both models can be considered good estimators since they are capable of generalize well the test set. The test phase evinced how the SVM is able to predict secondary structure conformation better than the GOR model with a 4% disparity for the Q3 score. This is a foreseeable outcome given that the GOR method is based on the assumption of statistical independence of all the 17 window residues while the SVM, thanks to its ability to handle large dataset, accounts for the correlations between neighbouring residues. Moreover, the GOR method performance could increase including all the GOR V improvements such as the the introduction of the decision constants in the final prediction of the conformational state, the inclusion of residues pairs and triplets and a smaller window of 13 residues (Sen *et al.*, 2005). Another important limitation imposed on both models implemented in this project is the bypassing of essential biological properties. Indeed, we arbitrarily defined only 3 states but ideal helices and sheets do not exist and there is not a precise boundary between the various conformations. Moreover, the described methods do not consider that helices involve hydrogen bonds of at least 4 neighbouring residues while sheets forms also thanks to hydrogen bonds between distant residues. Hence, another source of error could be the incapacity of these secondary structure prediction methods in considering non-local interactions: the important inter-sequence interactions are short ranged and only one-fourth of the total information needed to determine the secondary structure is available from local inter-sequence correlations (Crooks and Brenner, 2004).

4 Conclusion

The GOR model and the SVM proved to be both relatively good predictors, giving about 64% and 69% of Q3 score respectively. However, the higher performance of SVM is highly linked to an increase of the time complexity. While the training phase of the GOR model took a couple of seconds, the SVM one was a matter of days. State-of-the-art secondary structure prediction methods have reached an accuracy of about 84% by means of powerful machine learning and deep learning techniques, however the theoretical accuracy limit of 88% (Rost, 2001) could be obtained by capturing non-local interactions between residues that are close in the 3D space but far away in the primary structure.

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.

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., and Bourne, P. E. (2000). The protein data bank. *Nucleic acids research*, **28**(1), 235–242.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, **2**(3), 1–27.

Chou, P. Y. and Fasman, G. D. (1974). Prediction of protein conformation. *Biochemistry*, **13**(2), 222–245.

Consortium, U. (2019). Uniprot: a worldwide hub of protein knowledge. *Nucleic acids research*, **47**(D1), D506–D515.

Crooks, G. E. and Brenner, S. E. (2004). Protein secondary structure: entropy, correlations and prediction. *Bioinformatics*, **20**(10), 1603–1611.

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.

Fox Naomi, K. and Brenner Steven, E. (2014). Chandonia john-marc. scope: Structural classification of proteins–extended, integrating scop and astral data and classification of new structures. *Nucleic Acids Res*, **42**(1), D304–D309.

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.

Gunn, S. R. *et al.* (1998). Support vector machines for classification and regression. *ISIS technical report*, **14**(1), 5–16.

Jones, D. T. (1999). Protein secondary structure prediction based on position-specific scoring matrices. *Journal of molecular biology*, **292**(2), 195–202.

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.

Kloczkowski, A., Ting, K.-L., Jernigan, R., and Garnier, J. (2002). Combining the gor v algorithm with evolutionary information for protein secondary structure prediction from amino acid sequence. *Proteins: Structure, Function, and Bioinformatics*, **49**(2), 154–166.

Magnan, C. N. and Baldi, P. (2014). Sspro/accpro 5: almost perfect prediction of protein secondary structure and relative solvent accessibility using profiles, machine learning and structural similarity. *Bioinformatics*, **30**(18), 2592–2597.

Malkov, S., Zivkovic, M. V., Beljanski, M. V., and Zaric, S. D. (2005). Correlations of amino acids with secondary structure types: connection with amino acid structure. *arXiv preprint q-bio/0505046*.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., *et al.* (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, **12**, 2825–2830.

Rost, B. (2001). Protein secondary structure prediction continues to rise. *Journal of structural biology*, **134**(2-3), 204–218.

Rost, B. and Sander, C. (1993). Improved prediction of protein secondary structure by use of sequence profiles and neural networks. *Proceedings of the National Academy of Sciences*, **90**(16), 7558–7562.

Sen, T. Z., Jernigan, R. L., Garnier, J., and Kloczkowski, A. (2005). Gor v server for protein secondary structure prediction. *Bioinformatics*, **21**(11), 2787–2788.

Spencer, M., Eickholt, J., and Cheng, J. (2014). A deep learning network approach to ab initio protein secondary structure prediction. *IEEE/ACM transactions on computational biology and bioinformatics*, **12**(1), 103–112.

Steinegger, M. and Söding, J. (2017). Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, **35**(11), 1026–1028.

Ward, J. J., McGuffin, L. J., Buxton, B. F., and Jones, D. T. (2003). Secondary structure prediction with support vector machines. *Bioinformatics*, **19**(13), 1650–1655.

Yang, Y., Gao, J., Wang, J., Heffernan, R., Hanson, J., Paliwal, K., and Zhou, Y. (2018). Sixty-five years of the long march in protein secondary structure prediction: the final stretch? *Briefings in bioinformatics*, **19**(3), 482–494.

5 Supplementary material

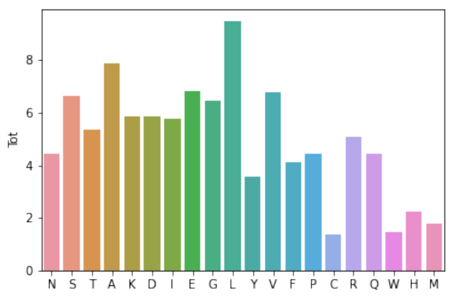

Fig. S1. Bar plot that shows the amino-acidic composition of the blind test dataset, obtained using the pandas package.

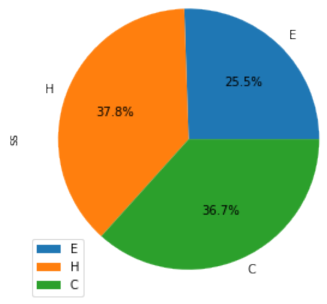

Fig. S2. Pie plot that shows the percentage of secondary structure conformations of the blind test dataset, obtained using the pandas package.

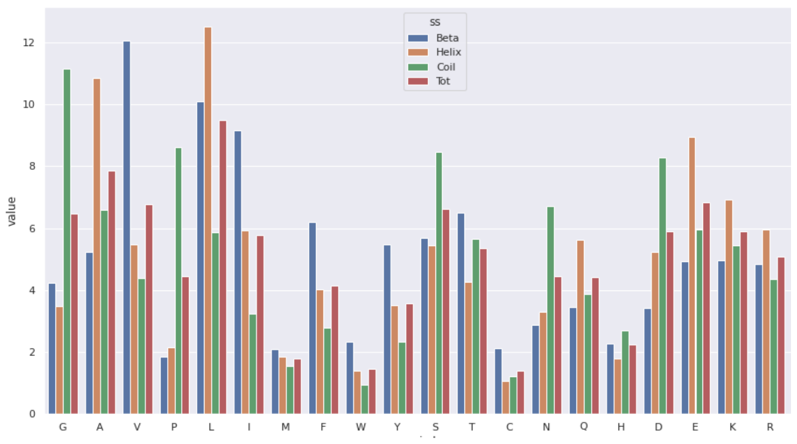

Fig. S3. Bar plot that shows the percentage of secondary structure conformations for each amino acid type of the blind test dataset, obtained using the seaborn package.

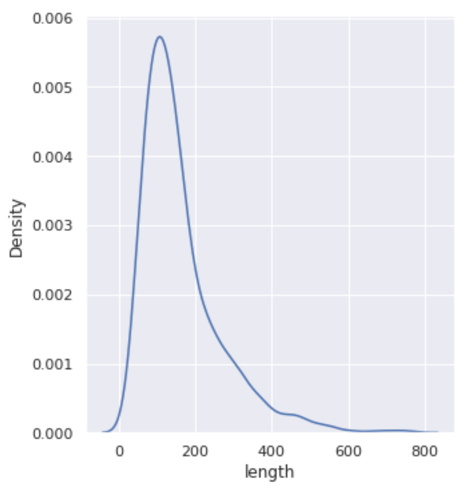

Fig. S4. Probability density distribution of the sequences length of the training dataset, obtained using the seaborn package.

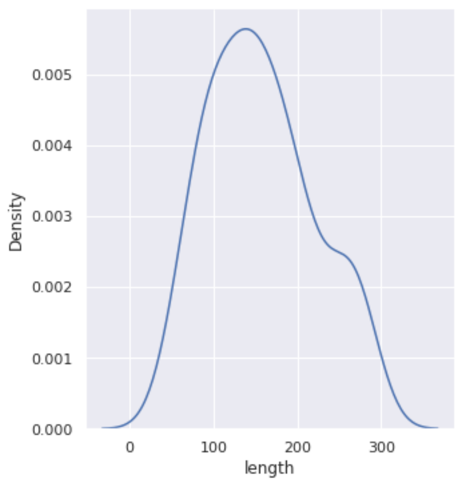

Fig. S5. Probability density distribution of the sequences length of the test dataset, obtained using the seaborn package.