Predicting secondary structure of proteins: a comparison between GOR method and Support Vector Machines

Becchi Tommaso

University of Bologna, Master Degree Course in Bioinformatics

Abstract

Motivation: One of the main biological problems is that of predicting the tertiary structure of a protein starting from the primary one. Nowadays the number of sequences is constantly increasing, while the number of tertiary structures is growing more slowly. The prediction of the secondary structure could help solve this problem: in this project, two methods for secondary structure prediction, GOR and SVM, are compared. For both methods, the training and the testing on a blind set are performed.

Results: GOR and SVM methods were correctly implemented: using them it is possible to predict the secondary structure of a protein starting from the primary sequence. The two methods have a different three-class accuracy: 62% with GOR and 70% with SVM. Despite this evident accuracy difference, the differences between the parameters that describe the performance of the two methods in predicting the 3 different secondary structures are less clear.

Availability: https://github.com/TommasoBecchi/lb2-2020-project-Becchi

Contact: tommaso.becchi@studio.unibo.it

Supplementary information: Supplementary data are available at https://github.com/TommasoBecchi/lb2-2020-project-Becchi

1 Introduction

The importance of proteins has always been known, as well as the fundamental role of their structure in determining correct functionality. For this reason, today one of the main goals of biology is to be able to determine the conformation of proteins in a very precise way.

Nowadays most of the structures are experimentally obtained thanks to X-ray crystallography[1]. However, if we compare the number of entries in the main databases, we observe a significant difference: about 150000 protein structures are present in PDB[2], while Uniprot contains more than 560000 manually curated sequences (Swiss-Prot)[3]. This is due to the many problems that arise during crystallography, starting with the high cost and the amount of time required.

Being able to predict the tertiary structure starting from the primary sequence is a problem still to be solved that can become a valid alternative to experimental methods.

Primary sequences contain only the symbols corresponding to the aminoacids and they are not very informative to understand protein folding. The secondary structures, on the other hand, are more useful because each secondary element corresponds to a specific folding in the tertiary structure. α-helices, β-sheets and coils identify functional domains and provide information about conformation constraints[4].

Since 1960, many methods for the prediction of the secondary structure have been developed. “First-generation” methods correlate the role of a single residue with its corresponding secondary structure. The most famous of these methods is the Chou-Fasman method[5][6] which is based on the relative frequency of each aminoacid in each secondary structure starting from a set of solved protein structures. The parameters obtained from this set are used to predict local secondary structure elements from primary sequences. The accuracy of these methods is around 50-60%.

“Second-generation” methods consider not only the propensity for a single residue, but they study the propensity for a residue also considering the adjacent ones. GOR method[7][8] uses a sliding window and predicts the probability to find a specific residue given the secondary structure of the central one. The accuracy of this method is slightly more than 60%.

“Third generations” methods introduce a new idea that leads to more accurate results: these methods use evolutionary information derived from the alignment of multiple homologs sequences[9]. The most conserved regions in a multiple sequence alignment are usually those with biological significance and they identify functional domains. Starting from multiple alignments, it is also possible to individuate insertions, deletions and repeated patterns. With this information the methods can distinguish between exposed and internal domains and they can reach more than 70% accuracy[10]. PSIPRED[11] uses as input sequence profiles deriving from PSI-BLAST[12] output and it reaches more than 80%

accuracy. Many other methods use this idea and they obtain a high level of accuracy: SSpro, ACCpredictors[13] and Jpred[14].
Newer methods require very complex calculations and increasingly sophisticated algorithms (Markov Models, Neural Networks[15][16], Support Vector Machines[17]…). For this reason, machine learning techniques play a central role in predicting secondary structures.
This introduction presents some of the main methods and many more are available. Unfortunately, it is still not clear whether there is a method which is better than the others. However, it is important to compare them to know what the strengths and weaknesses of each of them are.
The goal of this project is to compare the performance of GOR and SVM[18] methods. This comparison starts with the training on sequence profiles obtained from multiple sequence alignments, followed by the testing. This step is performed by predicting the secondary structure of a "blind set" of 150 sequences.

2 Materials and Methods

2.1 Training dataset

JPred4 dataset[14] is used as a reference set for the training. He was obtained from a starter set with 1987 representative domain sequences from each superfamily in SCOP v2.04[19]. This dataset covers all possible three-dimensional folds and it also reduces the likelihood of trivially detectable sequence similarities.
Low-resolution data (≥2.5Å), domains with few residues (length<30) and data with missing DSSP[20] information for more than 9 consecutive residues are removed from the dataset. In the end, the dataset contains 1497 domain sequences: they were split into training (1348) and blind testing sets (149). In this project, only the training set was used to train the two methods.

2.1.1 Statistical analysis of the dataset

The distribution of domain lengths

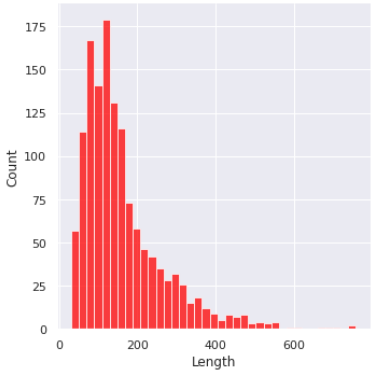

Figure 1. Distribution of domain lengths

The distribution in Figure 1 shows that most of the domains in the dataset have a length lower than 200 residues. Moreover, few domains have a length greater than 400 residues.

The relative abundance of secondary structure conformations

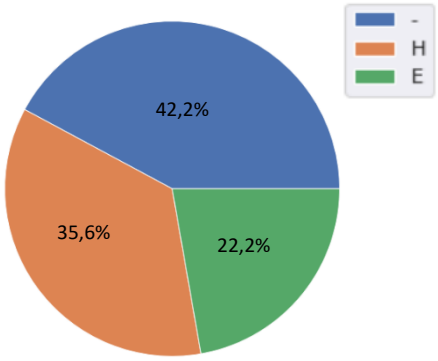

Figure 2. The relative abundance of secondary structure conformations. The symbol "- "means coil, "H" and "E" indicate helix and strand.

Figure 2 shows that the three classes are balanced in the dataset. Strands and helices maintain the same proportions compared to Swiss-Prot[21]. The number of coils is higher than in Swiss-Prot because in JPred this number sums all conformations other than beta and alpha.

Comparative amino-acid composition

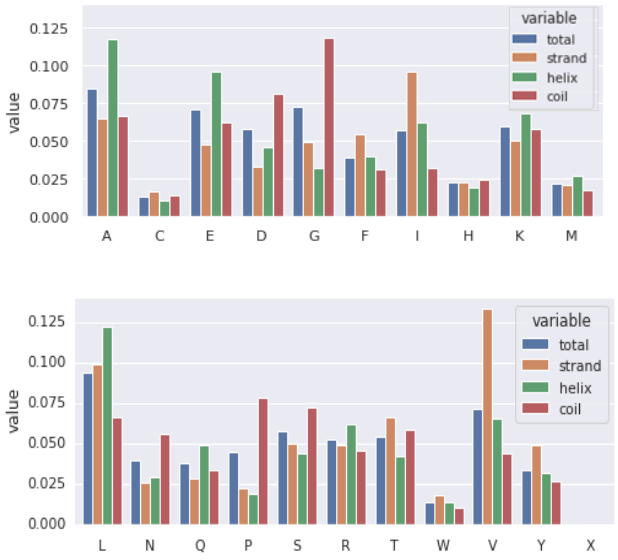

Figure 3. Comparative aminoacids composition. The relative frequencies of total residues, residues in strand, coils, or helix are shown for each type of residue. This graph allows us to observe which aminoacids have a particular propensity for a specific secondary structure in JPred4.

Different aminoacids have a different propensity for a specific secondary structure conformation. Figure 3 shows this trend for each residue. Glycine (G) and Proline (P) break the alpha-helices

and they are therefore found mainly in coils. Alanine (A), Leucine (L), Glutamate (Q), Methionine (M) and Lysine (K) prefer to adopt helical conformations. Isoleucine (I), Valine (V), Threonine (T) and aromatic residues (W, Y, F) are mostly found in strands. Therefore, the dataset is coherent with the chemical-structural properties of the residues.

Taxonomic classification

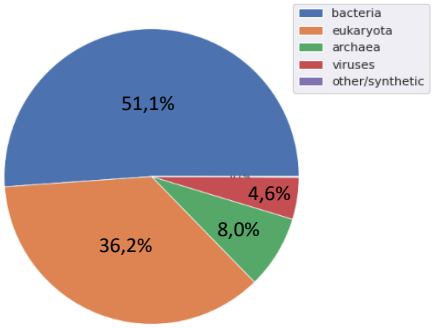

Figure 4. The relative abundance of taxonomic classes in the Jpred4 dataset.

Structural classification (SCOP class)

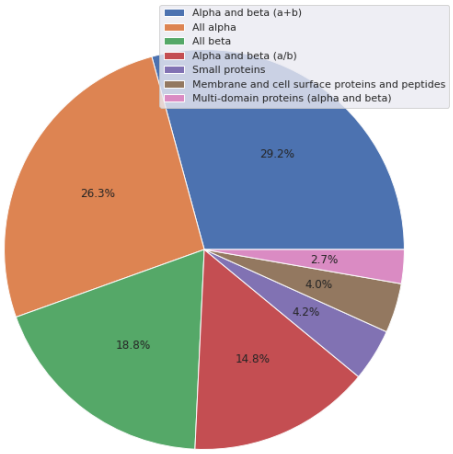

Figure 5. The relative abundance of SCOP classes in Jpred4 dataset

Figure 5 shows the relative abundance of each SCOP class. α+β class is the most abundant, followed by all-α class and all-β class. This is consistent with the proportion present in Swiss-Prot[21].

2.2 Blind test set description

Since JPred4 was used to perform the training, a new dataset is needed to evaluate the performance of the two methods. This is the so called "blind test set": it contains 150 sequences that reproduce the distribution of secondary structure of the training set. Cross-validation on the training set is not sufficient to estimate unbiased parameters: cross-validation may lead to some overfitting on training data, it necessary to use data never seen before.

2.2.1 Selection of proteins

From Protein Data Bank (PDB)[22] are extracted all the entries that respect the following conditions:

- Deposit after Jan, 2015
- High-quality experimental data: X-ray crystals with resolution < 2.5Å
- Chain length between 50 and 300 residues
- Only proteins, no RNA/DNA

The FASTA files of the sequences that meet these criteria are downloaded.

2.2.2 Internal redundancy reduction

To be sure that the final 150 sequences are sufficiently representative, it is necessary to reduce the internal redundancy. It is, therefore, necessary to cluster together the sequences that have a sequence identity greater than 30% and coverage greater than 50%. This step was performed with *blastclust*[23]. This tool identifies 3850 clusters, then a representative sequence is held for each cluster.

2.2.3 External redundancy reduction

There is a risk that some of the chosen sequences are similar to those present in the training set. This does not respect the need to have sequences never seen before.
To avoid this possible redundancy, it is necessary to use BLAST suite of programs[24]: the JPred4 database was formatted into a BLAST database with the *makeblastdb*[24] tool and the representative sequences were compared to this database with *blastp*[24].
The similarities between all the sequences of the two sets were obtained and all the sequences with a similarity greater than 30% were removed. 496 sequences are deleted. In the end, the set contains 3354 entities.
Among the remaining sequences, 150 were randomly chosen.

2.2.4 Generate DSSP file for each sequence

The PDB files were downloaded for each sequence in the blind set. The secondary structure sequence was inferred with DSSP[20][25]. This method implements an algorithm which calculates the most probable secondary structure assignments given the geometry of the protein's 3D structure. DSSP uses the hydrogen bonding pattern to discriminate between the possible secondary structures. The presence or absence of an H-bond can be detected by the bond energy (structures obtained with X-ray crystallography do not contain most hydrogen atom coordinates). DSSP defines eight possible types of secondary structure and in this project they are grouped as following:

- 3, 4, 5-turn helix: Helix

- Extended strands and residues in isolated β-bridge: Strand
- Turn, bend and no assignment: Coil

The secondary structure sequences were extracted from DSSP output files.

2.3 Sequence profile generation

Multiple sequence alignments allow to consider in the analysis also evolutionary information. A MSA contains the notion of protein family: all the sequences in the family are aligned using a substitution matrix. The results contain more information than the single sequences: conserved regions are more easily identifiable.
Sequence profiles are a condensed representation of a MSA obtained for a given target protein sequence. They are represented as a matrix of dimension L x 20: L is the length of the target protein sequence and 20 is the number of different residues. Pij indicates the frequency of residue j in the i-th position of the alignment.
UniprotKB/SwissProt[26] was used as a comparison set in which to search for sequences to align. First of all, this set must be formatted with *makeblastdb*[24].
In this project the profiles of both the training and the blind set were obtained with PSI-BLAST[12]. It performs multiple search iterations using a position-specific substitution matrix (PSSM) computed during the search: it is usually more sensitive than BLAST in finding distantly related sequences. In this project, for each sequence the algorithm performed a maximum of 3 iterations, the e-value threshold is 0.001 and number of alignments equal to 1000.
Some target sequences gave empty output or they did not present a PSI-BLAST output: this is due to the lack of similar sequences with which to generate the profiles. These sequences were removed from both training and blind set.
The correct profiles are:

- 1204 in the training set
- 120 in the blind set

2.4 GOR method

2.4.1 General overview of the method: parameter estimation

The Garnier-Osguthorpe-Robson (GOR)[8] is a method for protein secondary structure prediction. It is based on a combination between information theory and Bayesian statistics. This method is based on two main concepts:

- secondary structures depend on amino acids propensities
- the conformation of a given residue position is influenced by neighbouring residues

Starting from this idea, GOR derives propensity matrices from training data and three different propensity matrices are computed for helix, strand and coil.
In these matrices log-ratio scores indicates the influence of the sequence context to the conformation assumed by the central residue.
This method uses the following information function to predict the conformation a given reside:

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

P(S|R) is the condition probability of observing the secondary structure conformation S when the residues is R. P(S) is the marginal probability of observing the secondary structure conformation S.
Using the probability chain rule, it possible to rewrite the equation as:

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

P(R,S) is the joint probability of observing the residue type R in conformation S. P(R) is the marginal probability of observing a residue type R. All the parameters present in this equation can be estimated from the training data.

GOR method does not consider the residues as single entities: they are analysed in the context in which they are found. For this reason, the concept of sliding windows of width *w* is introduced. Therefore, the information function is extended over a *w* number of residues which have the central one as a reference.
It is impossible to evaluate all the necessary terms directly using the training dataset.
Therefore, the concept of independence assumption must be introduced: the residues $R_{-d},\ldots,R_d$ are statistical independent. The window-based information function is computed as the sum of individual single-residue functions:

$$I(S;R_{-d},...R_0,...R_d) = \sum_{k=-d}^{d} I(S;R_k)$$

The parameters are:

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

In this project the input are not single sequences but profiles: for each position, all possible frequencies must be considered

$$I(S;PW) = \sum_{k=-d}^{d}\sum_{R_k} PW[R_k] * I(S;R_k)$$

2.4.2 General overview of the method: prediction phase

Each residue of a query sequence is analysed: the predicted conformation of a residue R at position j is the one having the highest value of the window-based information function:

$$S^* = \text{argmax}_S\, I(S; R_{-d}, \ldots, R_d) = \text{argmax}_S \sum_{k=-d}^{d} I(S; R_k)$$

2.4.3 Implementation of the method

In this project, GOR method was implemented with two python scripts (one for the training and one for the prediction, they are available in the supplementary materials). The width *w* of the sliding window was set to 17. The input to estimate the parameters were the profiles and the corresponding DSSP files. The outputs were four 17x20 matrices: one for the helix, one for the strand, one for the coil and one with the overall frequencies.
Using these matrices, the prediction step calculates the most probable secondary structure for each residue.

2.5 Support Vector Machines

2.5.1 General overview of the method

Support Vector Machines (SVM)[18] are supervised learning models that are used both for classification and regression problems. The purpose of these algorithms is to identify the plane that best separates different classes (classification) or the one that best represents the relationship between different variables (regression). Compared to other machine learning methods, SVM has the advantage to be less prone to overfitting and it allows to distinguish between classes characterized by high feature space. This last property is due to the implementation of Kernel functions.

In this project SVM is used for a classification problem. In classification problems the best hyperplane is the one that maximizes the distance between the support vectors margins (the closest point for each class to the hyperplane) as shown in Figure 6.

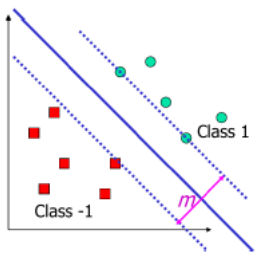

Figure 6. Schematic representation of a separating hyperplane in a 2D space. The best hyperplane is the one that maximizes the value of the margins *m*

The hyperplane is defined with the equation $w^Tx + b = 0$, it separates the space in two regions:
$w^Tx + b > 0$ and $w^Tx + b < 0$
Then for each training point the conditions are:

$$\mathbf{w}^\top \mathbf{x}_i + b \leq -\rho/2 \text{ if } y_i = -1$$
$$\mathbf{w}^\top \mathbf{x}_i + b \geq \rho/2 \quad \text{if } y_i = 1$$

These conditions can be summarized with the equation:

$$y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq \rho/2$$

The problem is therefore a quadratic optimization problem in which the goal is to maximize 2/||W||.
It is equivalent to the following

$$\begin{array}{lll} \text{minimize} & \frac{1}{2}\|\mathbf{w}\|^2 & \text{over } \mathbf{w} \\ \text{s.t.} & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 & \forall i \end{array}$$

To solve it, SVM uses a Lagrange multiplier that allow to obtain the dual problem

$$\begin{array}{lll} \text{maximize} & \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle & \text{over } \alpha \\ \text{s.t.} & \alpha_i \geq 0 & \forall i \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 & \end{array}$$

The parameters w, b that describe the separating hyperplane can be calculated as:

$$\begin{array}{lll} \mathbf{w} & = & \sum_s \alpha_s y_s \mathbf{x}_s \\ b & = & y_k - \sum_s \alpha_s y_s \langle \mathbf{x}_s, \mathbf{x}_k \rangle \qquad \text{for any } k \text{ s.t. } \alpha_k > 0 \end{array}$$

The classifying function for new points becomes:

$$y(x) = \sum_s \alpha_s y_s (x_s^T x) + b$$

If the points are not linearly separable there are two different situations. In the case of even a single outsider, the model introduces the so called “soft margins” idea: a slack variable describes the upper bound of allowed errors (Figure 7).

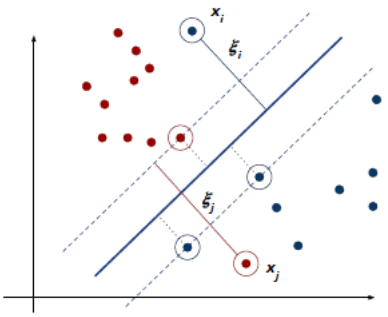

Figure 7. Schematic representation of “soft margins” idea. Some points on the wrong side of the margin are allowed and their distance is indicated with ξ

If the classes are not linearly separable, even with the implementation of soft margins, Kernels functions are used. The idea is that the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable.

2.5.2 Implementation of the method

In this project, the SVM method was implemented with the *libsvm*[26] package (v. 3.24), one of the most used SVM implementations. The algorithm must distinguish between 3 classes: helix, strand and coil. This is a multi-class SVM problem: the prediction is performed comparing the values of the three discrimination functions.
With this package two command-line programs were used:

- *svm-train*: takes in input a training file and produce a SVM model file in output
- *svm-predict*: takes in input a testing file and a trained SVM model and produces a prediction file in output.

In the *svm-train* command, the parameter *-t* was equal to 2. This parameter specifies the kernel type and the value 2 means that a radial basis function (exp(-gamma*|u-v|^2)) is selected.
The input data for classification problem with *libsvm* is required to be in the format:
<class> <feature_index>:<feature_value>
In this project, to produce the input data, each sequence profile was associated with the corresponding DSSP file. The class was registered as the secondary structure of the central residue for each window and each element of the 17x20 profile window was selected as the feature.
The values 1,2 or 3 was assigned to the class attribute, depending on the secondary structure (1 for H, 2 for E, 3 for C). The feature space presents at most 340 features: one for each frequency found in the profile window (0 values were removed).

2.5.3 Grid search

SVM algorithm is based on some main hyperparameters that must be optimized to find the ones that allow to obtain a better model. The training was performed four times on each set, changing on each iteration the γ parameter in kernel function (-g) and the cost parameter C (-c):

- g: 0.5, c: 2
- g: 0.5, c: 4
- g: 2, c: 2
- g: 2, c: 4

2.6 Evaluation procedure and scoring measure

SS prediction is a multi-class classification with 3 classes: helix, strand and coil. The evaluation of the model (GOR and SVM) performances can be computed using multi-class confusion matrix: the outputs for the two models are comparable then it is possible to use the same scoring indexes.

2.6.1 Multi-class confusion matrix

Comparing observed and predicted DSSP sequences, it is possible to compute a 3-class confusion matrix.

		PREDICTED		
		H	E	C
OBSERVED	H	p_{HH}	p_{HE}	p_{HC}
	E	p_{EH}	p_{EE}	p_{EC}
	C	p_{CH}	p_{CE}	p_{CC}

Table 1. Three-classes confusion matrix. The columns indicate the predicted secondary structure for each residue with a certain method (SVM or GOR). The rows indicate the observed secondary structure for each residue, obtained with DSSP. The values in the diagonal indicate good prediction: predicted and observed ss are the same for a specific residue

Three-class accuracy (Q3) is computed from this matrix as:

$$Q_3 = \frac{p_{HH}+p_{EE}+p_{CC}}{N}$$

This score gives you the overall accuracy of the model, meaning the fraction of the total samples that were correctly classified by the classifier.

For each of the 3 classes, it is possible to extract from the 3-classes confusion matrix a 2-class matrix like the following (the following explanations take class H as an example).

		PREDICTED	
		H	non-H
OBSERVED	H	$c_{H=}\ p_{HH}$	$u_{H=}\ p_{HE} + p_{HC}$
	non-H	$o_{H=}\ p_{EH} + p_{CH}$	$n_{H=}\ p_{EE} + p_{EC} + p_{CE} + p_{CC}$

Table 2. 2-class matrix

Using these 2-class matrices, for each class it is possible to calculate the correct and incorrect predictions:

- cH= helix residues correctly predicted in class H (correct positive)
- oH= non-helix residues predicted as H (over-prediction)
- uH= helix residues predicted as non-H (under-prediction)
- nH= non-helix residues predicted as non-H (correct negative)

These values were used to calculate the binary scores for each of the three classes:

- Sensitivity (true positive rate): measures the proportion of actual positives that are correctly identified as such
- Positive predicted value (precision): proportion of observed positive values over all the positive predictions
- Matthew's correlation coefficient: measures the correlation between observed and predicted data with a value between −1 and +1. A coefficient of +1 represents a perfect prediction, 0 no better than random prediction and

−1 indicates total disagreement between prediction and observation.

$$Sen_H = \frac{c_H}{c_H + u_H}$$

$$PPV_H = \frac{c_H}{c_H + o_H}$$

$$MCC_H = \frac{c_H \times n_H - o_H \times u_H}{\sqrt{(c_H + o_H) \times (c_H + u_H) \times (n_H + o_H) \times (n_H + u_H)}}$$

2.6.2 Cross-validation

Before using the entire training set to get the parameters, a cross-validation step on the training set was performed. This procedure allows to verify if the models have a consistent performance: the accuracy can vary significantly using different set as training and testing.

In this project the training set was divided in 5 sets. For each cross-validation, 4/5 of the sequences in the original training set were used as a new training set while the remaining ones were used for the testing.

After each CV run, it is necessary to compute performance scores on the current testing set.

Once the entire cross-validation procedure is completed the final value for each score is computed as the average value over the five CV runs. Average values must be accompanied by standard errors, defined as:

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

Where n is the number of samples and σ is the standard deviation.

3 Results

3.1 Cross-Validation results

After having completed the SVM cross-validations with the different parameters described above (section 2.5.3), it is possible to study the results to find out what the best combination is.

All the parameters chosen to describe the performance of the method indicate that the best combination is the one with γ equal to 0.5 and C equal to 2 (Table 3).

	γ =0.5	γ =2
C =2	70.82±0.28	46.74±0.36
C =4	70.21±0.28	46.65±0.35

Table 3. Average three-class accuracy resulting from the different CVs in the grid search. This parameter is used as an example to show which is the best combination but also the parameters which are not shown here (SEN, PPV and MCC) lead to the same conclusion (all the results of the grid search are available in the supplementary materials).
It is possible to observe that the results change little when C is modified while γ has a great influence on the performance.

Analysing the cross-validation results (Table 4) it is possible to observe that there are no significant differences in performance between the different partitions: the maximum standard error (considering both models) is equal to 1.07%.

SVM seems to perform better overall: only SEN_E, SEN_H and PPV_C have a higher value with GOR than with SVM.

All the MCC values and the three-class accuracy are significantly higher with SVM.

		GOR	SVM
SEN	**H**	80.27±0.51	69.17±0.96
	E	70.46±1.07	40.97±1.04
	C	43.55±0.84	87.80±0.32
PPV	**H**	63.26±0.67	84.82±0.46
	E	48.82±1.02	79.20±0.87
	C	80.05±0.48	62.43±0.39
MCC	**H**	0.525±0.003	0.658±0.007
	E	0.441±0.004	0.494±0.006
	C	0.417±0.004	0.492±0.004
Q3		62.52±0.16	70.82±0.28

Table 4. Cross.validation results for both GOR and SVM models. For SVM the results obtained with C=2 and γ=0.5 are shown. The table highlights the highest value obtained between the two methods for each parameter.

3.2 Blind test results

		GOR	SVM
SEN	**H**	75.82	65.13
	E	73.11	50.55
	C	46.04	88.46
PPV	**H**	65.36	86.96
	E	52.33	82.45
	C	75.43	60.28
MCC	**H**	0.514	0.643
	E	0.471	0.568
	C	0.414	0.499
Q3		63.40	70.84

Table 5. Blind test results for both GOR and SVM models. The table highlights the highest value obtained between the two methods for each parameter.

The performance of the models on the blind set shows very similar values to the cross-validation results (Table 5). The parameters with a higher GOR value are the same obtained previously: SEN_E, SEN_H and PPV_C.

3.3 Discussion

The results of the cross-validation on the training set confirms that JPred4 it is a good representative of the protein space: standard error is low for all parameters.

SVM method performs better than GOR method as anticipated in the introduction. GOR method is based on the assumption of statistical independence between the residues in the window: this simplification could be the cause of its lower performance. SVM

is able to consider the full correlation between neighbouring residues and the conformation of the central residue, thanks to its ability to handle large feature spaces with kernel functions.
The results obtained on the blind set are very similar to those obtained with the CV: this confirms the validity of the two methods and the correct choice of the sequences in the blind set.

4 Conclusion

In this project, two methods for secondary structure prediction were implemented and tested. The GOR method, using Bayesian statistics and the information theory, and the SVM method, a machine learning model for multi-class classification through the combination of binary classifiers. Both methods use evolutionary information in the form of sequence profiles.
GOR and SVM methods were able to predict the secondary structure of a set of proteins starting from their sequence profiles with a three-class accuracy (Q3) higher than 60%; the SVM method was able to reach 70%.
In the end, the SVM seems to be most suitable for the prediction of secondary structure.

References

[1] Smyth, M. S. & Martin, J. H. J. x Ray crystallography. *Mol Pathol* **53**, 8–14 (2000).
[2] https://www.rcsb.org/stats/
[3] https://www.uniprot.org/
[4] Rost, B. Review: Protein Secondary Structure Prediction Continues to Rise. *Journal of Structural Biology* **134**, 204–218 (2001).
[5] Chou, P. Y. & Fasman, G. D. Prediction of protein conformation. 24.
[6] Fasman, G. D. Conformational Parameters for Amino Acids in Helical, α-Sheet, and Random Coil Regions Calculated from Proteinst. 12.
[7] Robson, B. Analysis of the Code Relating Sequence to Conformation in Globular Proteins. 141, 15 (1974).
[8] Garnier, J., Gibrat, J.-F. & Robson, B. [32] GOR method for predicting protein secondary structure from amino acid sequence. in *Methods in Enzymology* vol. 266 540–553 (Academic Press, 1996).
[9] Cuff, J. & Bartonl', G. J. Evaluation and Improvement of Multiple Sequence Methods for Protein Secondary Structure Prediction. 12.
[10] Jones, D. T. Protein Secondary Structure Prediction Based on Position-specific Scoring Matrices. 8.
[11] McGuffin, L. J., Bryson, K. & Jones, D. T. The PSIPRED protein structure prediction server. *Bioinformatics* **16**, 404–405 (2000).
[12] Altschul, S. F. *et al.* Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res* **25**, 3389–3402 (1997).
[13] Magnan, C. N. & Baldi, P. SSpro/ACCpro 5: almost perfect prediction of protein secondary structure and relative solvent accessibility using profiles, machine learning and structural similarity. 6.
[14] Drozdetskiy, A., Cole, C., Procter, J. & Barton, G. J. JPred4: a protein secondary structure prediction server. *Nucleic Acids Res* **43**, W389–W394 (2015).
[15] Rost, B. & Sander, C. Improved prediction of protein secondary structure by use of sequence profiles and neural networks. 5.
[16] Rost, B. & Sander, C. Combining evolutionary information and neural networks to predict protein secondary structure. *Proteins* **19**, 55–72 (1994).
[17] Kim, H. & Park, H. Protein secondary structure prediction based on an improved support vector machines approach. *Protein Eng Des Sel* **16**, 553–560 (2003).
[18] Hsu, C.-W., Chang, C.-C. & Lin, C.-J. A Practical Guide to Support Vector Classification. 16.
[19] Andreeva, A., Kulesha, E., Gough, J. & Murzin, A. G. The SCOP database in 2020: expanded classification of representative family and superfamily domains of known protein structures. *Nucleic Acids Res* **48**, D376–D382 (2020).
[20] Kabsch, W. & Sander, C. Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers* **22**, 2577–2637 (1983).
[21] https://www.uniprot.org/statistics/Swiss-Prot
[22] https://www.rcsb.org/
[23] http://nebc.nerc.ac.uk/bioinformatics/documentation/blast/blastclust.html
[24] Camacho, C. *et al.* BLAST Command Line Applications User Manual. 23.
[25] Andersen, C. A. F., Palmer, A. G., Brunak, S. & Rost, B. Continuum Secondary Structure Captures Protein Flexibility. 10.
[26] https://www.uniprot.org/downloads
[27] Chang, C.-C. & Lin, C.-J. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.* **2**, 1–27 (2011).