Report

Models for signal peptide prediction: comparing Von Heijne algorithm and Support Vector Machines

Giacomo Orsini [1,*]

[1] Department of Pharmacology and Biotechnology, Alma Mater Studiorum - Università di Bologna, Bologna, Italy

[*] To whom correspondence should be addressed.

Abstract

Motivation: Signal Peptides (SP) play a crucial role in the secretory pathway of cells. In-silico prediction of signal peptides offers a valuable solution to the biological problem of identifying proteins implied in this pathway, a significant task for scientific efforts such as drug discovery. Many computational methods for prediction have been proposed in literature, with different features and performances. In the hereby presented experiment, two predictors for signal peptides were constructed using the Von Heijne algorithm and the machine learning method of Support Vector Machine. A Cross Validation step has been added in both workflow. The models were trained and tested using a datasets of reviewed proteins, containing examples of both signal peptides endowed proteins and proteins without it. The performances of the two final models were compared.
Results: Performances evaluation has shown that the SVM model is overall a better classifier with respect to the Von Heijne model.
Contact: giacomo.orsini2@studio.unibo.it
Supplementary information:

1 Introduction

Signal peptides (SP) play a crucial role in cellular processes, serving as guiding tags for proteins destined for secretion or integration into cellular membranes. These short amino acid sequences facilitate the accurate trafficking of proteins within cells, ensuring they reach their intended destinations. Immature proteins directed toward the cell secretory pathway (ER-Golgi-Membrane-Extracellular) are endowed with a signal sequence in the N-terminal region; the signal sequence is cleaved after the protein reaches its final destination. 5 types of signal peptides exist, depending on the signal peptidase that cleaves them and their pathways (SEC or TAT).

Recognition of signal peptides is a crucial step for the characterization of protein function and subcellular localization, as well as for deciphering cellular pathways. Moreover, knowledge of protein localization allows to identify potential protein interactors, providing insightful information for drug discovery. Despite their importance, it is challenging to determine whether unknown proteins have signal peptides due to the inherent characteristics of the SPs. These characteristics include the dissimilarities in domain properties, the absence of cleavage site conservation across proteins and species, and the resemblance of the hydrophobic core to other structures such as the transmembrane helix or the transit peptides.

The signal peptide has an average length of 16-30 residues. It has a modular structure comprising three separate regions with different properties: a positively charged region at the N-terminal (1-5 residues long), a hydrophobic core similar to a transmembrane helix (7-15 residues), and a polar C-terminal domain (3-7 residues). The cleavage site is locate at the C-terminal and it has an A-X-A motif weakly conserved. The cleavage site allows for the cleavage of the signal peptide to make the protein mature.

The biological question of predicting the presence of secretory signal peptides in proteins is a relevant problem of protein function and subcellular localization prediction that has been explored extensively in scientific literature.

1.1 Signal peptide prediction

Despite their biological importance, experimentally identifying signal peptides can be challenging and resource-intensive. This has led to the development of computational methods, which offer a more efficient and cost-effective approach, opening a new set of challenges. Available in-silico methods can be divided into three main classes:

- Homology-based approaches. In these approaches, information is transferred from closely related sequences. They can be accurate but require a good template, experimentally studied.
- Algorithmic and probabilistic approaches. A traditional algorithm takes some input and some logic in code and drums up the output.

In this category, methods such as Hidden Markov Models and Weight matrix approaches can be found.
- Machine learning approaches. A Machine Learning Algorithm takes an input and an output and gives the logic that connects them, which can then be used to work with new input to give one an output. Typical Machine learning methods are Artificial Neural Networks (including recent deep-learning methods), Decision trees and Random Forests, and Support Vector Machines.

The problem of signal peptide prediction (and more in general motives prediction) has two dimensions: the correct identification of protein containing the motif from proteins not containing it and the labelling of the motif inside the protein sequence.

1.2 Tools for signal peptide prediction

1.2.1 Von Heijne algorithm

The first method specific for signal peptide prediction was introduced by Gunnar Von Heijne in 1983 (Von Heijne, 1983). The von Heijne method was based on a reduced-alphabet weight matrix combined with a rule for narrowing the search region; only seven different weights at each position, corresponding to groups of AAs with similar properties were estimated manually (rather than using automatic procedures). The matrix score aimed to recognize the location of the SP cleavage site. The scores were computed for positions 12-20 counted from the beginning of the hydrophobic region (Defined as the first quadruplet of AAs with at least three hydrophobic residues). In the first benchmark the procedure correctly predicted 92% of sites on data used to estimate it, while on a later benchmark the test performance was only 64% (a sign of overfitting).

Since its first publication, the Von Heijne method was updated by introducing a more structured weight matrix (that uses log odds scores), pseudo counts to mitigate the sampling error (being the fact that the model is constructed from a limited set of examples), and a longer window of residues for the search of the cleavage site (40-50 residues) (Von Heijne, 1986).

This updated version of the method was used in the experiment (with a window of 90 residues).

1.2.2 McGeoch approach

In 1985, the first feature-based approach was published (McGeoch, 1985), based on different combinations of sequence derived features that could be able to discriminate Signal peptides from other proteins. Two discriminatory features were identified: the length of the uncharged region, from the first uncharged residue after the 11th residue to the next charged residue, and the maximal hydrophobicity in a window of 8 residues, calculated 18 positions downstream the from the start of the uncharged region. No cleavage-site prediction was implemented. The model was shown to be quite accurate for predicting signal peptides in eukaryotes, with an accuracy of around 90%. However, it was less accurate for predicting signal peptides in prokaryotes. This is because prokaryotic signal peptides are more diverse than eukaryotic signal peptides.

1.2.3 Machine learning methods

1.2.3.1 Neural Networks After the increase of interest in this machine learning method in the 80s, Neural Networks (NN) have affirmed themselves as the most successful framework for predicting SPs and their cleavage site. Early approaches used a shallow networks approach (few hidden layers): a fixed-size portion of the N-terminus (e.g. the first 20 residues) was given as input to the NN; a sliding window approach was implemented to scan both for the presence of the signal peptide and cleavage site position (identification and labelling). Tools that used this approach are SignalP1-4 (Nielsen *et al.*, 1997; Nielsen and Krogh, 1998; Bendtsen *et al.*, 2004; Petersen *et al.*, 2011) and SPEPLip (Fariselli *et al.*, 2003).

Novel approaches have progressively evolved toward the use of more complex and deeper network architectures. Examples of tools are DeepSig (Savojardo *et al.*, 2018) and SignalP5-6 (Almagro Armenteros *et al.*, 2019; Teufel *et al.*, 2022). These deep learning methods have nowadays affirmed themselves as gold standards; for example SignalP5-6 has been demonstrated to be capable of predicting all 5 types of signal peptides.

1.2.3.2 Support Vector Machines Support Vector Machines (SVM) are a powerful machine learning algorithm that can be used for classification and regression. Briefly, SVMs work by finding a hyperplane in the feature space that separates the data points into two classes with the largest possible margin. This hyperplane is then used to classify new data points. Although less populat than NN, SVMs have been used to predict signal peptides with great success. SVMs have been shown to be very accurate at predicting signal peptides (Vert, 2001).

1.3 Experimental design and goals

In the present work, two different methods for model creation have been used to face the biological challenge of signal peptide prediction. The two models generated with the different methods have been compared to find the best one. A proper dataset of proteins comprising signal peptide proteins and non-signal peptides (positives and negatives) has been constructed starting from data fetched on the Swiss-Prot database; to avoid biases, a data preprocessing step with numerous quality checks has been performed. The dataset has been divided into two sets, the Training set and the Benchmarking set; the Training set has been further divided into 5 Cross Validation sets (CVs).

The first method used has been the Von Heijne algorithm, an algorithmic method based on position-specific matrices. Briefly, the steps to implement this method have been:

- Training of 5 different models using different Cross Validation subsets combinations.
- Extraction of an optimal threshold to discriminate positives and negatives by testing the models on a validation set.
- Performance evaluation of the models on a testing set.
- Average optimal threshold computation and training of a final model using the entire Training set.
- Performance evaluation of the final model tested on the Benchmarking set, with the average threshold as discriminatory.

The second method used has been the machine learning algorithm of Support Vector Machines in the form of a binary classifier (SVC). Briefly, the steps to implement this method have been:

- Discriminatory feature extraction and input encoding of data
- Training of different models using different combinations of hyper-parameters, features and Cross validation sets.
- For each feature combination, the best model selection is by testing the models on validation sets, saving the most common hyper-parameters and the best scoring set of Training sets.
- Performance evaluation of the best model on the testing set.
- Selection of the best set of features and creation of a final model, trained on the Training set and tested on the Benchmarking set.

The final results, as well as the partial results of the two methods, have been compared. A standard set of metrics has been used for the performance evaluation. Finally, eventual misclassifications (False Negatives and False Positives) have been examined.

The experiment has been designed as explained to answer two main questions: the biological question of signal peptide prediction, with the

creation of a well performing predictive model, and the comparison of different models to find the best one. Although these were the final goals, the project has been focused more on the design and implementation of the two methods than the results themselves.

The results have shown that, while both models make misclassification errors, the SVC performs better than the model created with the Von Heijne method, remarking the strength of a well-implemented machine learning method compared to an algorithmic one. Both models, with different performances, were generally able to discriminate the signal peptide proteins.

2 Methods

2.1 Datasets

Protein sequences, codes and features for the creation of a dataset have been retrieved from the SwissProt database (Release 2023_04) with two separate query searches: one to fetch reviewed proteins endowed with the signal peptide (positives) and another to fetch the ones not containing it (negatives). Quality checks and filtering procedures have been carried out to process the raw data, creating two clean datasets: the Training and the Benchmarking sets. The Training set has been divided into 5 smaller Cross Validation (CV) sets to perform a Cross Validation procedure. Detailed statistical analysis on secondary data and metadata has also been carried out.

2.1.1 Data retrieval

All data have been retrieved from the UniProtKB SwissProt database (Release 2023_04). This database contains manually reviewed protein sequences, each with standard sets of information about their characteristics and evidence, allowing the user to access and retrieve such entries with query searches easily. To fetch proteins containing and not containing the signal peptide, the Advanced Search Tool has been used: two separate query searches have been carried out on the website, each with its specifications, but both starting with the same premises. Sequences coming from the Archaea and Bacteria domains have been excluded, leaving only the ones belonging to the Eukarya domain; moreover, only reviewed proteins with more than 30 residues have been selected, with 30 being the upper bound of the average number of residues of a Signal Peptide.

The output of each fetching has been stored into two preliminary datasets in TSV format for subsequent data preprocessing procedures: the *Preliminary Positive set* and the *Preliminary Negative set*.

Hereby are the specifics for the queries:

1. Preliminary Positive set:
   - Taxonomy: Eukaryotic sequences.
   - Reviewed: only reviewed sequences.
   - Length: sequences greater or equal to 30 residues long.
   - SP evidence: only entries with experimental SP evidence.
2. Preliminary Negative set:
   - Taxonomy: Eukaryotic sequences.
   - Reviewed: only reviewed sequences.
   - Length: sequences greater or equal to 30 residues long.
   - SP: only entries without an SP (at any evidence level).
   - Cellular location: proteins with experimental evidence of being localised in the cytosol, nucleus, mitochondrion, plastid, peroxisome or the cell membrane.

Hereby, the information retrieved in TSV format:

1. Preliminary Positive set:
   - Signal peptide.
2. Preliminary Negative set:
   - Subcellular location.

The *preliminary Positive set* comprised 2,969 entries, while the *preliminary Negative set* 31,619.

2.1.2 Quality checks

Positive and Negative raw data retrieved in the previous step suffered from biases that must be corrected with a proper preprocessing procedure. Specifically, in this preliminary step, data has been filtered out to eliminate misclassification, inadequate sequences and redundant proteins.

2.1.2.1 Filtering

From the preliminary Positive set, the entries with a non defined signal peptide (marked with a "?" symbol) and the ones with a peptide smaller than 13 residues have been removed, as the length of the cleavage site of a signal peptide is 13 residues. From the preliminary Negative set, to make sure that positive proteins have not been misclassified as negatives, entries with a subcellular location in the Endoplasmic Reticulum, Golgi Apparatus, Lysosome or secreted have been removed, as signal peptides can be found in these organelles and have this property.

After this filtering step, the *cleaned Negative set* contained 30,011 entries and the *cleaned Positive set* 2,942. The UniProt IDs have been extracted from the cleaned dataset. The IDs have been used to retrieve the protein sequences in FASTA format using the Uniprot ID Mapping Tool. From these new sets of data, the *Filtered Positive* and *Filtered Negative datasets* have been created.

2.1.2.2 Redundancy

The query search procedure from SwissProt ensures the fetching of all proteins with the desired features; hence, the chance of having proteins with similar or identical sequences is very high. This leads to a potential redundancy bias that can harm the creation of predictors. To ensure the absence of redundancy in the Positive and Negative sets, the MMSeqs2 program has been used to cluster proteins of both sets separately and extract representatives. The IDs of the representatives have been extracted from the outputted FASTA files and saved in TXT files, one for the Negative set, now containing 9523 IDs, and one for the Positive, with 1093 IDs. The IDs of each set have been randomized to be used to create the Training and Benchmarking sets.

Hereby are some specifications about the MMSeq2 program: MMSeq2 (Many-against-Many sequence searching) is a software suite to search and cluster huge protein and nucleotide sequence sets. It supports different clustering modes and is faster than other methods, such as BLAST (reference:). A single run produces many files, among which two are important: cluster results_rep_seq.fasta, a FASTA file containing all the representative sequences, one for each found cluster, and cluster-results_cluster.tsv, reporting the IDs of each sequence and their cluster's representatives. Some particular options have been applied in the experiment to run the jobs:

- Percentage sequence identity threshold: 30% sequence identity.
- Length coverage threshold: 40% coverage.
- Coverage computation mode: 0 (coverage of query and target).
- Clustering mode: 1 (Connected component, i.e. single linkage).

2.1.3 Training and Benchmarking sets

The non-redundant, filtered, randomized data gathered in the previous step has led to the construction of the Training and Benchmarking sets.

- Training set: it is used to train the methods, optimize model hyperparameters and perform Cross Validation experiments.
- Benchmarking set: the holdout dataset is used to test the generalization performance of the different models.

These two datasets are crucial to the whole project, as all the predictive computational methods are rooted in unbiased and well balanced sets of data. Proper pairs of such datasets should contain an equal proportion of negative and positive examples; moreover, the proportion of whole examples between Training and Benchmarking should be 80:20.

To meet such requirements, both TXT files containing the IDs of the Positive and the Negative sets have been split into 2 subsets with an 80:20 proportion: the splits containing 80% of the entries have been concatenated in the *Training set*, while the splits containing the remaining 20% of the proteins have been united in the *Benchmarking set*.

The newly created datasets in TXT format contained just the UniProt IDs. The UniProt ID Mapping Tool has been used separately on both sets to generate a TSV file containing numerous useful metadata:

- Signal peptide length
- Protein length
- Taxonomic lineage (to extract Kingdom)
- Organism (to extract species)
- Protein sequence

Extra "shell" steps allowed for extracting the desired information and adding an extra column containing the class of each protein in binary form (0 negatives and 1 positives). The final TSV files had the following columns:

- Entry name
- Signal peptide length (NaN if not present)
- Protein length
- Kingdom
- Species
- Protein sequence
- Class

The Training set and Benchmarking set, now suited for being analyzed and further manipulated, contained 8492 and 2124 entries, respectively.

2.1.4 Cross validation sets

A Cross Validation procedure is crucial to evaluate the model design and avoid potential bias such as overfitting; the validation of the model on a limited amount of data allows to estimate how well the model will generalize, in a less optimistic way, also helping in the choice of the hyperparameters.

To maintain the correct proportion of negative and positive entries in each Cross Validation set, the Training set has been divided into a *parsed Positive Training set* (874 entries) and a *parsed Negative Training set* (7618 entries). Each of the two sets has been then divided into 5 subsets, leading to 5*Cross Validation positive* subsets and 5 *Cross Validation negative* subsets. The positive and negative Cross Validation subsets have been then paired and concatenated into 5 Cross Validation sets, containing respectively: *CV0, CV1, CV2, CV3* 1699 entries and *CV4* 1697 entries.

2.1.5 Statistical analysis

Statistical analysis has been carried out on the Training and Benchmarking sets to extract useful information about the proteins, for the contextualization of the problem and to double check for potential left out biases. The metadata retrieved from UniProt comprised information on the taxonomic lineage of the protein, the length of the protein and the signal peptide (if present) and the protein sequence. Specifically, the analyses have been the following:

- Signal peptide length distribution (Fig: 1, Supplementary Table S1). The length has been normalized to a $\log_{10}$ scale.
- Protein length distribution (Supplementary Figure S1 and S2 and Supplementary Table S2). The length has been normalized to a $\log_{10}$ scale.
- Taxonomical classification (Supplementary Figure S3, S4 and S5).
- Amino acidic composition of the signal peptides compared to a background SwissProt distribution (amino acidic frequencies considering all the entries in the database), also considering the amino acid properties (Fig: 2) (UniProtKB/Swiss-Prot, 2023).
- Sequence logos of the cleavage site (with the use of the online tool WebLogo) of the signal peptide (Fig: 4, Fig: 3).

Regarding the Signal peptide length distribution, no significant differences can be appreciated between the Training set and Benchmarking set: the mean length is around 23 residues in both cases, the smaller and the larger signal peptides from both sets have a comparable length (Supplementary Table S1) and the distributions overall have a similar shape (Fig: 1). Considering also the protein length distribution, it has been seen that the protein length of positive proteins follows a different distribution with respect to the negative ones in both sets (Supplementary Figure S2 and S3), in particular, positive proteins have a mean length of 398 residues in the Training set and 368 in the Benchmarking set, while the negatives have a mean length of 546 and 549 respectively (Supplementary Table S3 and S4). For what concerns the compositional statistics of the signal peptides, the amino acidic composition has resulted comparable between the two sets, but different from the background SwissProt distribution: in particular, a noticeable increase in the presence of leucine, alanine and methionine, as well as a slightly increased presence of valine, cysteine and serine has been found, while tyrosine, asparagine, glutamine, histidine, aspartic acid, glutamic acid, lysine and arginine residues are under represented (Fig: 2). Considering the amino acidic properties (Supplementary Figure S6), the observations can be explained with an increase in the presence of hydrophobic residues and a decrease in the presence of the charged ones; this can be explained by the fact that one of the conserved regions of a signal peptide is the hydrophobic core, the residues making this core, in the context of the signal peptide, are more represented that the charged ones, even if they too are conserved and have an essential role, in particular at the N terminal where usually there is a positively charged region.

The sequence logos of the cleavage sites also confirm this. The cleavage site motif (residues -13 to +2) have been extracted to compute the logos of signal peptides in the Training and Benchmarking sets: a noticeable presence of conserved leucines can be seen at the N.termini of the motif, making the hydrophobic core, while alanine is also present in the form of an A-X-A or V-X-A motif. On the contrary, charged residues are not conserved in this important motif of the signal peptides (Fig: 4, Fig: 3).

Regarding the taxonomic analysis, results state that the majority of the proteins of both sets came from the Metazoa Kingdom, followed by Fungi and Viridiplantae (Supplementary Figure S3), while at the species level, the most represented one was *Homo sapiens*, followed by *Saccharomyces cervisiae* and either *Arabidopsis thaliana* in the Training set (Supplementary Figure S4) and *Schizosaccharmyces pombe* in the Benchmarking set (Supplementary Figure S5).

2.2 The Von Heijne method

The first method applied for constructing a model for signal peptide prediction has been the Von Heijne algorithm. The Von Heijne algorithm is a computational method for prediction based on weight matrices applied in literature to predict signal sequences and trans-membrane segments.

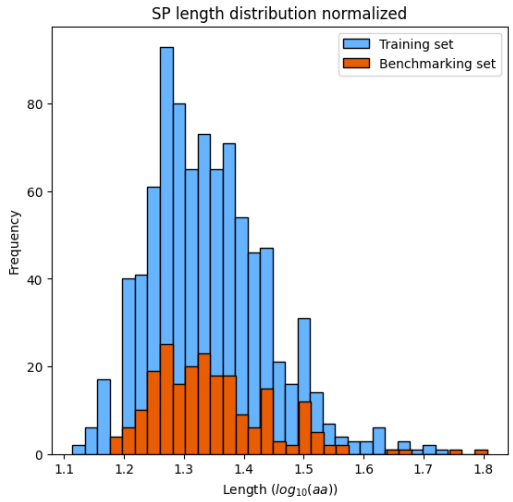

Fig. 1. Normalized length distribution of Signal peptide proteins in both Training and Benchmarking sets

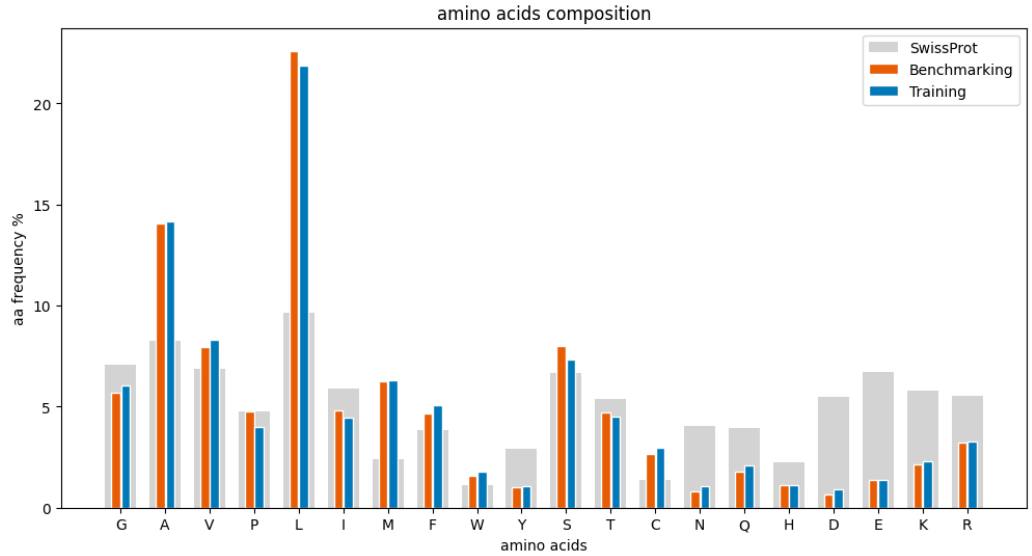

Fig. 2. Amino acidic distribution. For each amino acid the frequencies of occurrence in each set (Training, Benchmarking and SwissProt) are compared.

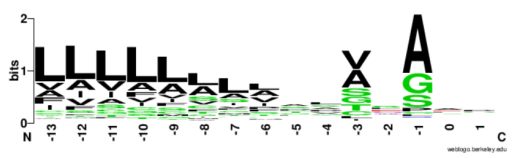

Fig. 3. Sequence logo of the cleavage site of Benchmarking set proteins.

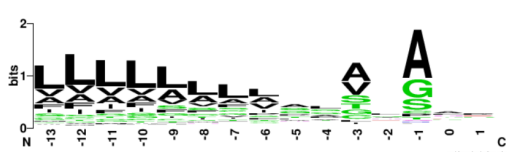

Fig. 4. Sequence logo of the cleavage site of Training set proteins.

In general, it offers a reasonable level of accuracy (60-80%). This method allows the detection of an SP in a given sequence and, with the

implementation of a sliding window approach, to label it, identifying its exact position in the sequence.

2.2.1 Principles

The algorithm workflow is the following: starting from a set of aligned sequence fragments, a position-specific probability matrix (PSPM) is computed. This matrix stores the frequency of each residue type at each position; the number of rows is equal to the number of different characters in the alphabet (20 for proteins) and the number of columns is equal to the length of the motif. From the PSPM, a position-specific weight matrix (PSWM) is computed. This matrix is identical in structure, but it stores the log-odds ratio between amino acid frequencies per position in the PSPM and a background model (the SwissProt composition). Finally, by choosing any fragment of a sequence, the log-likelihood can be computed starting from the PSWM, returning a score that indicates the likelihood of occurrence of the motif. In this way, proteins can be scanned to search for the motif with a sliding window approach.

Given a set S of N aligned sequences of length L, the PSPM (M) is computed as follows:

$$M_{k,j} = \frac{1}{N}\sum_{i=1}^{N} I(s_{i,j} = k) \tag{1}$$

Where:

- k is the residue corresponding to the k-th row in the matrix.
- $s_{i,j}$ is the observed residue of aligned sequence i at position j.
- $I(s_{i,j} = k)$ is an indicator function (1 if the condition is met, 0 otherwise).

From the PSPM (M), the PSWM (W) is computed as follows:

$$W_{k,j} = \log\frac{M_{k,j}}{b_k} \tag{2}$$

Where:

- b_k is the frequency of residue type k in the background model.

In the PSWM, a value $W_{k,j}$ can be positive when the probability of having a certain residue in a given position differs from the background and is higher ($M_{i,j} > b_k$). Contrarily, when it is lower or equal to 0, the probability of having that residue differs from the background, and it is lower ($M_{i,j} \leq b_k$); it is more likely that the position is a random site rather than a functional one.

In order to avoid zero probabilities in the PSPM and hence the impossibility of computing the log-odds ($\log(0)$ is undefined), pseudocounts are added during the computation of PSPM. In the simplest setting, the count matrix is initialized, assuming each residue is observed at least once in all positions. Practically, this means initializing the PSPM with 1 in all cells, while formally, the formula for computing the PSPM (M) becomes:

$$M_{k,j} = \frac{1}{N+20}(1 + \sum_{i=1}^{N} I(s_{i,j} = k)) \tag{3}$$

Given any piece of sequence $X = (x_1, .., x_L)$ of length L, one can compute the log-likelihood score of X given the PSWM as:

$$score_{(X|W)} = \sum_{i=1}^{L} W_{x_i,i} \tag{4}$$

2.2.2 Implementation

In the present experiment, the Von Heijne method has been used to create a predictor for the presence of signal peptides in proteins. Considering the datasets previously generated and the introduction of a Cross Validation step, the workflow has been organized in the following phases:

1. Training, Cross Validation and threshold selection: in 5 different runs, 3 Cross Validations sets have been used to compute a PSWM (model training). For each run, the model has been tested on a validation set to extract an optimal threshold, used then to discriminate positive and negative proteins in a final testing procedure, on which performance evaluation has been done.
2. Prediction: the average of the thresholds obtained at each of the 5 runs has been calculated. A new training procedure has been performed with all the positive examples of the Training set. The newly generated PSWM has been tested on the Benchmarking set, and the average threshold has been used to discriminate positives and negatives. The performance of the final model has been evaluated, and the results were stored in a file.

2.2.2.1 Training, Cross Validation and threshold selection

In this first part, 5 different models have been trained and tested using the Cross Validation sets to extract an optimal threshold for discriminating signal peptide proteins (positives) and non signal peptide proteins (negatives). 5 runs have been done, and at each run, 3 positive Cross Validation sets (sets containing just positive examples) have been used for training, 1 whole (containing both negatives and positives) Cross Validation has been used for validation, and 1 whole Cross Validation set has been used for testing.

1. Training: alternatively, 3 of the 5 Cross Validation positive sets have been used to compute a PSWM matrix (model training). First, the cleavage motifs have been extracted from the sequences (position -13 +2 to the signal peptide length). From this stacked alignment and the SwissProt distribution, the PSPM and the PSWM have been computed. For each run, the PSWM has been stored in a TXT file for a total of 5 PSWMs.
2. Cross validation: the first 90 N-terminal positions from the protein sequences of the validation subset have been extracted. A sliding window approach has been implemented to calculate the score (log-likelihood) of every 15 residues subsequence for each sequence (subsequence 1-15, subsequence 2-16 ...). The highest positional score for each sequence has been saved as the global score. Precision and recall values have been computed using the precision-recall curve and used to compute the F1 score (Equation 27). The optimal threshold of the model was the one associated with the best F1 score.
3. Testing: again, the first 90 N-terminal positions from the protein sequences of the validation subset have been extracted, and the score has been calculated using a sliding window approach on 15 residue subsequences, scoring the highest score for each sequence. This time, if the score was higher than the optimal threshold found in the previous step, the protein was predicted as positive (1) and if the score was lower, as negative (0). Using the lists of true classes and predicted classes, each model's performance was evaluated with different metrics.

Finally, a TSV document containing all the results from each run has been saved.

2.2.2.2 Prediction

After obtaining the results of each model and their metrics, the thresholds used in each run have been averaged to obtain an average threshold that then has been used to discriminate protein in a final model trained on the whole positive Training set and tested on the Benchmarking set.

1. Training: all the positive examples from the Training set have been used to train a final model. The cleavage sites have been extracted, and the PSPM and PSWM have been computed as described before.
2. Testing: considering the whole Benchmarking set, the first 90 N-terminal residues have been extracted, and the score has been computed using the same sliding window approach. This time, the average of the threshold obtained in the previous phase has been used to discriminate the proteins as Negatives and Positives. In the end, by using the list of True classes and the newly predicted ones, scoring metrics have been used to evaluate the performance of the final model obtained with the Von Heijne method.

2.3 SVM

The second method for constructing a model for signal peptide predictions has been the machine learning algorithm of Support Vector machines (SVM).

2.3.1 Principles

Support vector machines are supervised machine learning algorithms for classification and regression tasks. In classification problems, the algorithm of SVM aims to find a hyperplane in an N-dimensional space (where N is the number of features) that distinctly separates a set of data points belonging to different classes. In a two dimensional space, the hyperplane is a line. The hyperplane is a decision boundary. The idea of the SVM is to find the hyperplane that best separates the classes, the one that is both equidistant from the closest examples of the two classes and that maximizes the distance between them, hence maximizing the so called margin. The maximization of the margin is the optimality criterion. The closest points of the two classes fall onto the borders of the margin and are called Support Vectors. Once the hyperplane is determined, new data can be classified by determining on which side of the hyperplane it falls. Given a binary classification case (Figure), where the Training set $D = \{(x_i, y_i)i = 1, 2, .., n\}$ comprises n points in the d-dimensional space ($x_i \in \mathbb{R}^d$), and $y_i \in \{-1, 1\}$ is a vector that contains the classes, the formal equation of the separating hyperplane (Figure 5) is:

$$w^T x_i + b = 0 \tag{5}$$

The points will belong to one class (+1) if $w^T x_i + b > 0$ or to the other (-1) if $w^T x_i + b < 0$. Many linear separator can be computed, but, as we said, the algorithm will search for the one that maximizes the margin. Given a distance d between a point x_i and the separator is:

$$d = \frac{w^T x_i + b}{||w||} \tag{6}$$

Not all the points are essential for the calculation, in fact the only points for which the distance is calculated are the closest to the separator, the support vectors. The distance between the support vectors belonging to the two different classes, the margin (ρ), has to be maximised: it is the optimality criterion. A new set of equations can be written considering the margin:

$$w^T x_i + b \leq -\rho/2 \text{ if } y_i = -1 \tag{7}$$

$$w^T x_i + b \geq \rho/2 \text{ if } y_i = +1 \tag{8}$$

Therefore, $y_i(w^T x_i + b) \geq \rho/2$ for any class. For support vectors, x_s, this inequality constraint is transformed into an equality one, such that $y_i(w^T x_s + b) = \rho/2$, as they lie on the line. Considering the distance equation (Equation 6)and the margin definition, the margin can be redefined as:

$$\rho = \frac{2}{||w||} \tag{9}$$

The maximization of such expression can be solved by minimizing the following expression:

$$\frac{1}{2}||w||^2 \tag{10}$$

With condition for all points:

$$y_i(w^T x + b) \geq 1 \tag{11}$$

The solution of this quadratic problem involves constructing a dual problem where a Lagrange multiplier α_i is associated with every inequality constraint in the primal (original) problem:

$$maximize \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \quad over\ \alpha \tag{12}$$

$$s.t. \qquad \alpha_i \geq 0 \qquad \forall i \tag{13}$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \tag{14}$$

Given a solution for all α_i, where $\alpha_i > 0$ for support vectors x_s, the solution of the original problem is:

$$w = \sum_s \alpha_s y_s x_s \qquad b = y_k - \sum_s \alpha_s y_s \langle x_s, x_k \rangle \tag{15}$$

for any k given that $\alpha_k > 0$.

2.3.1.1 Margins

Regarding the type of margin, there can be two conditions depending on the problem: if data is linearly separable, SVM is used with a hard margin; when using hard margins no misclassification is allowed as no point is allowed between the margin. The formulas shown in the previous paragraph describe the hard margin condition.

If data is not linearly separable or some misclassification is allowed to generalize better, the classifier is used with a soft margin. A misclassification error is introduced, and the algorithm will find the hyperplane that maximizes the margin and minimizes the misclassification error. Formally, a so called Slack Variable ξ is introduced so that the margin maximization function is:

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

With condition for all points:

$$y_i(w^T x_i + b) \geq 1 - \xi_i \tag{17}$$

C is a hyperparameter that can control overfitting, as it decreases the relative importance of the margin maximisation and the fitting of the data, minimising the errors on the training points. A high value of C will result in a narrow margin, decreasing the possibility of a training error. In contrast, a small C will result in a wider margin, increasing the possibility of a training error. the solution of the quadratic problem is always the Equation 12 but now the value of α_k has an upper-bound given by the value of C: $C \geq \alpha_k \geq 0$.

2.3.1.2 Kernels

In the case of non linearly separable data, a method can be implemented on SVM that allows for the linear separation of the classes: Kernels. The principle of kernels function is to transform the data into a space in which the data becomes linearly separable, usually a space with higher dimensions. To avoid subsequent computation in high dimensional space that would be much complicate, the kernel trick is used: the kernel function is a function that takes the input vectors as input and gives the scalar product of the vectors in the feature space. This scalar product can then be substituted in the objective function, providing a solution to the now linearly separable problem with not increased complexity in the computation. Formally, the transformation of the points is:

$$\phi(x_i \rightarrow \phi(x)) \tag{18}$$

And the scalar product becomes:

$$K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle \tag{19}$$

The quadratic problem equation becomes:

$$maximise \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \quad over\ \alpha \tag{20}$$

There exist lots of different Kernel functions, such as the linear kernel, the polynomial kernel, the sigmoid kernel. For the implementation of the method in the experiment, the RBF Kernel (Radial Basis Function) has been used.

A Gaussian Kernel is used to perform the transformation when there is no a priori knowledge about the data. In the RBF kernel is a Gaussian kernel in which the radial basis method has been added to improve the transformation. The Gaussian Kernel function is:

$$K(x_i, x_j) = exp[\frac{-||x_i - x_j||^2}{2\sigma^2}] \tag{21}$$

In the RBF Kernel, $1/2\sigma^2$ becomes γ, so that the function is:

$$K(x_i, x_j) = exp[-\gamma||x_i - x_j||] \tag{22}$$

γ is a hyperparameter that decides how far the influence of a single training example reaches during transformation, which in turn affects how tightly the decision boundaries end up surrounding points in the input space. If there is a small gamma value, points farther apart are considered similar. Larger values of gamma cause points to be closer together. Considering the inverse relation between γ and σ, the standard deviation in the Gaussian function, a small value of σ corresponds to higher values of γ and vice versa.

2.3.2 Implementation

To apply this machine learning method to our problem and create a support vector classifier (SVC) for signal peptides prediction, the following steps have been carried out:

1. Feature Extraction and Input encoding. Discriminatory features have been selected and encoded into a proper format for the SVM algorithm.
2. Training, validation and testing. Using a grid search for hyperparameters, many models have been trained on Cross validation sets, tested on a validation set to select the best one, and evaluated on a testing set.
3. Prediction. By choosing the hyperparameters of the best model, a final model has been trained on the Training set and tested on the Benchmarking set.

The Scikit Learn library has been used to implement this method in Python. It is a Machine-Learning library based on Numpy, Scipy and Matplotlib.

The Kernel type chosen for the classifier has been the RBF Kernel, which requires the definition of two hyperparameters:

- γ parameter. As previously stated, it decides how far the influence of a single training example reaches during transformation. It can be described as the inverse of the radius of influence of the support vectors. The behaviour of the model is very sensitive to the γ parameter: if γ is too large, the radius of the area of influence of the support vectors only includes the support vector itself, and the model will overfit the data, no matter the values of C. When γ is very small, the model is too constrained and cannot capture the data's complexity or "shape".
- C parameter. As previously stated, the C parameter trades off correct classification of training examples against maximization of the decision function's margin. A smaller margin will be accepted for larger values of C if the decision function is better at correctly classifying all training points. A lower C will encourage a larger margin and, therefore, a simpler decision function at the cost of training accuracy. In other words, C behaves as a regularization parameter in the SVM.

2.3.2.1 Feature extraction and input encoding

An SVM requires input examples in the form of D-dimensional vectors. The dimensions of such vectors are equal to the data's number of features. To optimally design the SVC, it is crucial to first define a set of discriminatory features for the proteins (the positive and negative examples) and encode them in the proper format for the algorithm. Therefore, the discriminatory features chosen have been:

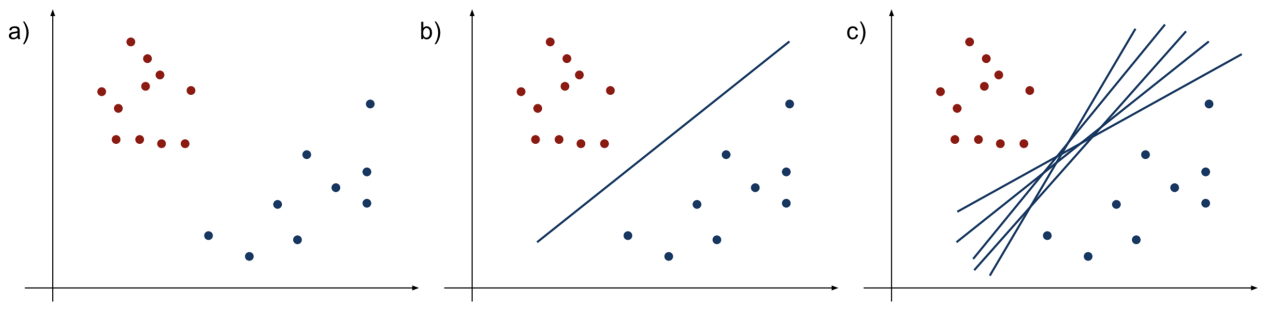

Fig. 5. Binary classification example. a) Training set separated in a R^2 feature space. b) Best performing linear separator. c) Other functional linear separator. Adapted from LB2 course slides.

- Amino acidic composition (C): as we have seen in statistical analysis, the frequency of some residues in signal peptides differs from the background SwissProt distribution.
- Hydrophobicity (HP): as we have seen, hydrophobic residues are more abundant in signal peptides. Hydrophobicity has been calculated using the Kyte & Dolittle scale (Kyte and Doolittle, 1982).
- Charge (CH): as we have seen, in the signal peptides' N-terminal region, a positively charged region is usually conserved.
- α-helix tendency (AH): as the hydrophobic amino acids tend to form a single α-helix, this feature may be discriminatory for signal peptides. For the calculation of α-helix tendency, the Chou and Fasman scale has been used (Chou and Fasman, 1978).

A "K" number of residues has been selected for each Cross Validation set sequence, starting from the N-termini. Considering each of the aforementioned features, except the composition, the score for each residue in the subsequences of length K has been computed using a sliding window approach. The sliding window was set to be 5 residues (7 for the α helix tendency) to include in the computation of the score for each residue the effect of its surrounding residues. Computations have been made thanks to the features scales retrieved from the Expasy database.

After calculating the scores for each residue, for each subsequence the following measurements have been computed:

- Maximal value
- Average value
- Position of the maximal value in the sequence

These values represent the final set of extracted features from the Hydrophobicity, charge and α-helix tendency discriminatory features.

The length of the subsequences K is a hyperparameter, and it was chosen to be in a range of 20 to 24 residues as the average signal peptide length. Alternatively, for every value of K, all the features have been extracted and scaled from 0 to 1 for all the proteins in each Cross Validation set and saved in TSV files, for a total of 25 TSV files (one for each CV and all K values). In each TSV file, the rows contained protein codes and the columns the features in the following order: the frequency of each amino acid, average value, maximum value, position of maximum value of the discriminatory features (in the order: hydrophobicity, charge and helix tendency) and finally the class of the proteins.

Storing the feature extraction results in such a format file allowed for an easy input encoding, as columns can be chosen to allow for different feature combinations, and TSV files can be easily transformed in pandas data frames and Numpy arrays. All the different combinations of features and hyper-parameters have been used to train the SVC.

2.3.2.2 Training, validation and testing

In this phase, numerous models have been trained on 5 different combinations of 3 Cross Validation sets, considering different combinations of extracted features and hyperparameters for each.

A grid search procedure had to be defined for combining the different hyperparameters. In a grid search procedure, a set of possible values for each hyperparameter is defined and all the possible combinations are tested for each CV run; the models are tested on a validation set, and the combination achieving the highest performance on the set is selected as the optimal one for that specific CV run, and fixed fort testing. The final value of each hyperparameter is the one that is most frequently selected during Cross Validation.

The range of possible values for the hyperparameters for the RBF Kernel have been the following:

- K=20, 21, 22, 23, 24
- C = 1, 2, 4, 8
- γ = 0.5, 1, 2, "scale"

Considering all the hyperparameter combinations, the 8 possible discriminatory feature combinations, and 5 different combinations of Cross Validation sets, the models trained with this method have been 3200. Considering a single feature combination, the models were 400. Considering then a single Cross Validation sets combination, the models were 80; the computation of 80 models defined a single run.

As done before with the Von Heijne algorithm, here also the Cross Validation sets (this time the whole CV sets, not just the positives) have been alternated to train, validate and test so that data coming from 3 sets has been used for training, accordingly with the hyperparameters selected in the grid search, 1 for validation to choose the best hyperparameters, and one for testing to evaluate the performance.

Each of the 80 models created in a run has been tested on the validation set, the performance has been stored temporarily, and the model that obtained the best MCC (Equation 23) has been saved (hence, the hyperparameters were stored). This model has then been tested on the testing set, and the performance saved. For each combination of discriminatory features, 5 runs have been done. To obtain the best model for the combination, the MCC of the 5 models has been averaged, and as hyperparameters, the most frequent ones were taken.

In the end, a TSV document containing the best model's performance for each feature combination was created.

2.3.2.3 Prediction

To train the final model, the combination of features that led to the model with the best performance and its hyperparameters have been chosen. Taking these settings, a final model was created, trained on the Training set (hence, trained on all the CV sets) and tested on the Benchmarking (encoded in the same way). The performance has been stored in a file.

2.4 Performance measurements

Standard sets of metrics for performance evaluation are normally chosen to evaluate binary classifiers, such as the one created with the Von Heijne algorithm and the machine learning SVM algorithm. These metrics are useful both to individuate the best model and to identify problems in the design of the model itself. These kinds of metrics are all based on the computation of the so called confusion matrix: in this 2x2 matrix, one dimension represents the observed class (positive or negative) and the other one the predicted class (positive or negative). This defines 4 categories of data: positive examples that have been classified as positives are the True Positives (TP); negative examples that have been classified as negatives are the True Negatives (TN); positive examples that have been classified as negatives are False negative (FN) and negative examples classified as positives are the False positives (FP). From these values, many metrics can be computed. Hereby is a list of the metrics used in this experiment:

- Matthew correlation coefficient (MCC). It is used to measure the difference between the predicted values and the actual values. It is a well balanced metric that ultimately tells how good the model is. It is defined as:

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{(\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)})} \tag{23}$$

- Accuracy (ACC). It computes the proportion of correct predictions among the total number of predictions. It is defined as:

$$ACC = \frac{(TP+TN)}{(TP+TN+FP+FN)} \tag{24}$$

- Precision. It is used to compute the ratio between the True Positives and the set of all the entries predicted as positives. With Recall is used

to compute the F1 score. It is defined as:

$$Precision = \frac{TP}{(TP + FP)} \tag{25}$$

- Recall. It is used to compute the ratio between the True Positives and the set of all that belong to the positive class. With Precision, is used to compute the F1 score.

$$Recall = \frac{TP}{(TP + FN)} \tag{26}$$

- F1 score. It is the harmonic mean of Precision and Recall, which balances Precision and Recall in binary classification. It is a metric used to evaluate the goodness of the method. Compared to the MCC score, it suffers more from class imbalance; therefore, the latter is preferred. It is defined as:

$$F1 = \frac{(Recall \times Precision \times 2)}{(Recall + Precision)} \tag{27}$$

- ROC curve and Area Under The ROC Curve (AUC). A ROC curve is a graphical plot that illustrates the performance of a binary classifier model at varying threshold values. It plots the true positive rate (TPR $\frac{TP}{(TP+FN)}$) against the false positive rate (FPR $\frac{FP}{(FP+TN)}$). AUC measures the entire two-dimensional area underneath the entire ROC curve from (0,0) to (1,1); briefly, when given one randomly selected positive instance and one randomly selected negative instance, AUC is the probability that the classifier will be able to tell which one is which.
- Standard error. It is a measure for telling how much a certain statistic differs in the different runs. Given the standard deviation σ and n number of samples, it is defined as:

$$SE = \frac{\sigma}{\sqrt{n}} \tag{28}$$

3 Results

3.1 Cross validation results

In Table 3.1 , the results of Cross Validation are shown for both the methods applied to create a predictor for Signal peptides. Regarding the Von Heijne method, as mentioned in the previous chapter, at each run, a model was trained on three positive Cross Validation sets. It was then first tested on a validation set to extract the optimal threshold (at which the F1 score is higher). This threshold was used to discriminate positive and negative proteins in the final testing step. The performance obtained on the testing set was evaluated and stored. In Supplementary Table S5, the performance of all the 5 models can be appreciated. The overall performance of the model was obtained by averaging the MCC score with standard error SD. The optimal thresholds were averaged to an average threshold.

As for the results, the MCC score of the models ranged between 0.627 (run 1) and 0.723 (run 4). The average MCC value is 0.679±0.013. This not-so-high MCC score is the result of a significant presence of both False negatives and False positives.

Regarding the SVM method, models have been trained considering different combinations of input features; for each feature set, 5 runs were made by training the models on 3 training Cross Validation sets with a grid search for hyperparameters. The models were tested first on a validation set to find the best one (highest MCC); this model was then tested on a testing set storing the performance. From each 5 runs, the average MCC was calculated and the most frequent hyperparameters were stored to define the best model for the feature combination. Supplementary Table S6 stores the extended results for these models.

As for the results, it can be seen how some combination of features led to different performances. In particular, the model obtained using Hydrophobicity, Charge and Composition as discriminatory features has obtained the best MCC, of 0.81 ± 0.009. Parameters of the model were C=4, γ=1 and k=23. The worst MCC has been obtained by the model trained only with compositional values (0.76 ± 0.017), while the model obtained from the combination of all the features has not performed as great as the aforementioned one (0.809 ± 0 .011). From these partial results, it can be seen that the α-helix tendency is not a good discriminatory feature for signal peptides, as models obtained with this feature consistently obtained worse results with respect to the ones that didn't have it.

Also in this case, a considerable amount of samples have been misclassified.

Comparing these partial results of Von Heijne and SVM, it is already evident that, on absolute, the machine learning method has been able to generate models that better classify the data presented.

3.2 Benchmarking results

The final results of the Von Heijne and SVM methods are reported in Tab **??**. For Von Heijne, results have been obtained by training the a PSWM using the positive Training set and testing it on the Benchmarking set, using as threshold the average threshold computed during the Cross Validation procedure. The final MCC of 6.9 highlights missclassification problems. Indeed, in later analysis, numerous False Negatives and False positives have been found. The extended results can be found in Supplementary Table S7.

The final results of the SVM method, seen in Table 3.2, have been obtained by training an SVM on the whole Training set and testing it on the Benchmarking set, using the hyperparameters and the feature set that gave the best results in the previous phase. The MCC of 0.821 suggest a good performance for this method applied to this problem. Although the performance of the classifier has been sufficiently good, a discrete amount of False Negatives and False Positives has been misclassified. und. The extended results can be found in Supplementary Table S8.

Overall, the final results have shown that, the predictor built with SVM gives better performances than the one built with the Von Heijne method. These results confirm what has already been explored in literature. Machine learning methods such as SVM and Neural Networks are more robust to noisy data, can be scaled to handle huge amounts of data and can handle non-linear relationships between the features and the output (for example with the use of Kernels in SVMs). Moreover, compared to the Von Heijne method, SVM can hold large amounts of different discriminatory features rather than just the amino acidic composition; this has been shown to be an important characteristic as the best obtained model was made with composition, Hydrophobicity and Charge rather then just the composition. Finally, the Von Heijne method has been applied only on a small number of residues corresponding to the conserved cleavage site, while SVM can be and have been applied on a broader subset of residues.

The performances have been lowered by misclassification of numerous proteins, in the form of False Negatives and False Positives. This suggests either a fallacy in the model design, in the dataset creation or in the methods themselves.

3.2.1 False positives and false negatives

Resulting False positives and false negatives from both the Von Heijne and the SVM methods have been extensively studied, in order to understand the origins of these misclassifications. The confusion matrices of the predictions can be found in Supplementary Figures S7 and S8.

- False positives (FP): one source of error has been found in the presence of transmembrane proteins and transit peptides in the dataset. These

Table 1.

Model (features)	Average MCC $\pm$ SD
VH	0.679±0.013
SVM (Comp.)	0.76 ± 0.017
SVM (Comp.+HP)	0.809 ± 0.01
SVM (Comp.+CH)	0.782 ± 0.011
SVM (Comp.+AH)	0.776 ± 0.011
SVM (Comp.+HP+CH)	0.81 ± 0.009
SVM (Comp.+HP+AH)	0.803 ± 0.009
SVM (Comp.+CH+AH)	0.795 ± 0.007
SVM (Comp.+HP+CH+AH)	0.809 ± 0 .011

Results of the cross validation step for both Von Heijne method and SVM. The MCC is reported with standard error.

Table 2.

Method (features)	MCC	ACC	TN	FP	FN	TP
VH	0.69	0.941	1863	69	56	163
SVM (Comp.+HP+CH)	0.821	0.966	1862	42	30	189

Performance of the final models. Reported the MCC, the accuracy, and the sets of TN, FP, FN, TP.

two domains, TM and TP, are not signal peptides, but they have similar characteristics, in particular they too have an hydrophobic core; the presence of the hydrophobic core is according to the hypothesis what caused their misclassification as positives. Transmembrane domain can be found along the sequences and can be mono- or multi- domains, while transit peptides are found at the N-termini. An increased percentage of this type of proteins in the false positives set in respect to the true negatives set has been highlighted from the statistics of both methods (Supplementary Figure S9 and S10).

The steps to find TM and TP have been the following:

1. Identification of the False Positives set and True Negatives set
2. Retrieving of information about TP and TM with the UniProt id mapping tool (experimental evidence)
3. Labelling of the proteins as TP, TM or both. For Transmembrane proteins, just the domains found in the first K (23) residues have been considered in the SVM statistics and the ones found in the first 90 amino acids for the VH.

For the Von Heijne False positives set among 69 FP, 10 have been labelled as TM and 29 as TP. For the SVM False positives set, among 42 FP 3 have been labelled as TM, 26 as TP and one had both.

- False negatives (FN): a source for FN has been found to be a different amino acidic composition, for the Von Heijne method of the cleavage site, for the SVM of the of the first K residues of some sequences of the dataset.

  In the case of the Von Heijne method, this difference in the composition may be due by the presence of distantly related proteins. The sequence logo of the cleavage sites (made using WebLogo) of FN proteins has been computed and compared to the sequence logo of the positives examples of the Benchmarking set (Supplementary Figure S11); it can be seen that leucines in the hydrophobic core are less conserved in the FN, as well as the AxA motif; moreover, at the end of the cleavage site, the presence of more charged residue can be noticed.

  In the case of SVM, the different amino acidic composition can be explained by the fact that FN proteins have unusually long or short signal peptides. Because of this, the main discriminatory regions such as the hydrophobic core are shifted with the respect to their average position, resulting in a different amino acidic composition of the K long fragments and a misclassification from the model. By computing the length distribution of the signal peptides (Supplementary Figure S12), comparing the false negative and the true positives of the Benchmarking set with the positive example of the Training set, a different distribution has been noticed. The amino acidic composition is also different, as in FN leucines and alanine are less represented, while polar residues and charged residues (arginine in particular) are over represented (Supplementary Figure S13). It is a composition similar to the one of the FN of the VH method.

  The steps to analyze the false negatives have been:

  1. Identification of FN and TP sets for both methods.
  2. Extraction of the cleavage sites from the FN proteins sequences and sequence logo computing for the Von Heijne method.
  3. Extraction of the first K residues from the FN and TP of the Benchmarking set and the positives of the Training set. Computation of the amino acidic frequencies and comparison.
  4. From the aforementioned datsets, extraction of the SP length data and comparison.

Given these analysis, to obtain better results some changes in the experimental design could be made: the Transmembrane and Transit peptide proteins could be excluded from the training datasets; in this way, the model would be more capable of correctly classify them as negatives. For the false negatives, further study on the effect of the signal peptide length and the position of domains should be made; distantly related homologs are often difficult to classify, hence a more detailed study of these entries is crucial to figure out the solution to this problem.

4 Conclusion

The goals of the experiment were to construct a well performing model for signal peptide prediction implementing two different methods and to

compare their goodness. Two models have been constructed using the Von Heijne algorithm, based on position-specific weight matrices (PSWM), and Support Vector Machines, a machine learning method.

A dataset of positive (proteins known to contain an SP) and negative (proteins known to not contain an SP) examples has been retrieved from the SwissProt database. Positive and Negative examples have been proportionally separated into two sets, the Training set and the Benchmarking set. The Training set has been further divided into 5 Cross Validation subsets. These were used to make a Cross Validation run, where the subsets were used for training, validating and testing the model in turns.

The Von Heijne model was created by training a position-specific weight matrix with the signal peptide cleavage site of all the positive proteins of the Training set, and using a threshold (computed from the optimal thresholds of each CV run), that allowed to classify the proteins as positive (higher than the threshold) or negative (lower than the threshold). The final SVM model was trained on the Training set, using a set of optimal discriminatory features and hyperparameters determined in the Cross Validation procedure. The final SVM model was the one whose input encoded features corresponded to amino acidic composition, hydrophobicity and amino acid charge.

The performances of the model have been evaluated with a standard set of metrics, in particular, the Matthews Correlation Coefficient (MCC, Equation 23) has been the ultimate evaluator for the goodness of the model.

The final model obtained with the Von Heijne method, as well as the models obtained from the Cross Validation procedure, always resulted in a lower performance (lower MCC) with respect to the SVM models. The Von Heijne models were more prone to make misclassification errors, as confirmed by the presence of many False positives and False negatives predictions with respect to the SVM. None of the two final models were without these types of errors, and by studying the False Positives and False negatives, it has been shown that the firsts had been caused by the presence of proteins with similar characteristics of the Signal peptides in the dataset (transmembrane proteins and transit peptides), while the latter from a different amino acidic composition with respect to the typical composition of the signal peptides and cleavage site motifs.

Ultimately, the results unequivocally show that the SVC, the support vector classifier, leads to better overall performance compared to the Von Heijne method. This can be explained by the nature of machine learning methods which are in general more robust and scalable than the algorithmic ones. Indeed, in the experiment the SVM method has been used to generate a lot of different models starting from different input features, settings and sets. Moreover, SVM allow for inputting large sets of discriminatory features with respect to the Von Heijne algorithm.

There is room for improvement in predicting SPs. One way to do so would be to merge the two approaches, as SignalP does. By utilizing PSWMs from the Von Heijme algorithm for initial motif recognition and SVM models for classification, the strengths of both methods would be exploited. One possibility for enhancing the SVMs is to use different encoding techniques that could better separate the classes. Another approach would be to create models that consider the structure of the SP sequences or to combine bioinformatics with comparative genomics for more confidence in the results. Deep learning techniques like DeepSig (Savojardo *et al.*, 2018) could also boost the performance of SP prediction.

References

Almagro Armenteros, J. J., Tsirigos, K. D., Sønderby, C. K., Petersen, T. N., Winther, O., Brunak, S., von Heijne, G., and Nielsen, H. (2019). Signalp 5.0 improves signal peptide predictions using deep neural networks. *Nature biotechnology*, **37**(4), 420–423.

Bendtsen, J. D., Nielsen, H., Von Heijne, G., and Brunak, S. (2004). Improved prediction of signal peptides: Signalp 3.0. *Journal of molecular biology*, **340**(4), 783–795.

Chou, P. and Fasman, G. (1978). Prediction of the secondary structure of proteins from their amino acid sequence. adv enzymol 47: 45–148. *Chou4547Adv. Enzymol*.

Fariselli, P., Finocchiaro, G., and Casadio, R. (2003). Speplip: the detection of signal peptide and lipoprotein cleavage sites. *Bioinformatics*, **19**(18), 2498–2499.

Kyte, J. and Doolittle, R. F. (1982). A simple method for displaying the hydropathic character of a protein. *Journal of molecular biology*, **157**(1), 105–132.

McGeoch, D. J. (1985). On the predictive recognition of signal peptide sequences. *Virus research*, **3**(3), 271–286.

Nielsen, H. and Krogh, A. (1998). Prediction of signal peptides and signal anchors by a hidden markov model. In *Ismb*, volume 6, pages 122–130.

Nielsen, H., Engelbrecht, J., Brunak, S., and Von Heijne, G. (1997). Identification of prokaryotic and eukaryotic signal peptides and prediction of their cleavage sites. *Protein engineering*, **10**(1), 1–6.

Petersen, T. N., Brunak, S., Von Heijne, G., and Nielsen, H. (2011). Signalp 4.0: discriminating signal peptides from transmembrane regions. *Nature methods*, **8**(10), 785–786.

Savojardo, C., Martelli, P. L., Fariselli, P., and Casadio, R. (2018). Deepsig: deep learning improves signal peptide detection in proteins. *Bioinformatics*, **34**(10), 1690–1696.

Teufel, F., Almagro Armenteros, J. J., Johansen, A. R., Gíslason, M. H., Pihl, S. I., Tsirigos, K. D., Winther, O., Brunak, S., von Heijne, G., and Nielsen, H. (2022). Signalp 6.0 predicts all five types of signal peptides using protein language models. *Nature biotechnology*, **40**(7), 1023–1025.

UniProtKB/Swiss-Prot (2023). Uniprotkb/swiss-prot protein knowledgebase release 2023_05 statistics.

Vert, J.-P. (2001). Support vector machine prediction of signal peptide cleavage site using a new class of kernels for strings. In *Biocomputing 2002*, pages 649–660. World Scientific.

Von Heijne, G. (1983). Patterns of amino acids near signal-sequence cleavage sites. *European journal of biochemistry*, **133**(1), 17–21.

Von Heijne, G. (1986). A new method for predicting signal sequence cleavage sites. *Nucleic acids research*, **14**(11), 4683–4690.