Title: SVM buried versus exposed transmembrane β barrel residues status prediction

## Introduction

Support vector machine (SVM) is a supervised machine learning technique which means that data for training is supplied with correct classification. It is non-parametric so there is no assumption on how the data is distributed. The algorithm is trying to find the hyperplane to separate training examples and in turn, assign them to different classes with the highest possible accuracy. The number of classes is defined by user beforehand(Mountrakis, Im, & Ogole, 2011). One of easily accessible ways to use SVM is through scikit-learn library(Varoquaux et al., 2015). There are 4 main types of kernels in SVM provided by scikit-learn: linear, polynomial, rbf and sigmoid. These kernel types define the function used for separating hyperplane determination. Different kernels might give the best results depending on the underlying problem one is trying to solve.

Assessing the accuracy of the model is important for getting an estimate of its capabilities and also for model selection. One way of accuracy assessment is cross-validation(Kohavi, 1995). In this technique, the training database is split into training set on which algorithm is trained and test set which is only used for testing the performance of the model. This approach allow for generalization the accuracy of the model, as test set is simulating real problem(need citation).

Trans-membrane β-barrels are transmembrane proteins formed by antiparallel β-sheet forming a barrel shape structure. Two neighbour residues inside in this β-sheet are always pointing in the opposite directions forming the in/out pattern with reference to the centre of the barrel. The residues pointing outside will always be nonpolar, as they are facing nonpolar inner part of transmembrane protein and residues pointing inside will always be polar(Zvelebil, M., & Baum, 2007). There are very little β-barrels structures experimentally solved so far. Predicting transmembrane regions is also difficult since they lack recognizable features such as stretch of 15-30 consecutive hydrophobic residues or positive inside rule present in helical transmembrane protein(Singh, Goodman, Walter, Helms, & Hayat, 2011). Prediction of the exposure status of residues (buried or exposed) is important because of its possible applications in side-specific mutational studies and in channel engineering(Singh et al., 2011).

There are several different approaches for prediction the exposure status of Trans-membrane β barrel residues.

For SVM its around 75-80% - best approach seems to be HMM – paper references

TMBHMM – 83%

## Methods

* Dataset

The dataset used in this project consisted of 69 transmembrane β barrel non-homologous proteins. For each protein, the exposure status in given position was provided. The dataset was organized in repeating three line pattern: protein ID, protein sequence and exposure status in separate lines for each entry.

* Including evolutionary information – PSI-Blast profiles

In order to add evolutionary information which might improve the accuracy model, PSI-BLAST was used to generate PSSM for each protein in the dataset. Swissprot database was chosen as reference database for PSI-BLAST instead of UniRef90 in the interest of time, as it allowed to decrease the time necessary to perform this step drastically. E-value was set to 0.01 and number of iterations to 3. Obtained profiles were stored in subdirectory as separate files for each protein in database.

* Extracting features from the dataset

For this purposed 3 separate lists were created one for storing protein ID, one for PSSM profiles and one for exposure status. To each list, related line from dataset file was appended. List of exposure status had to be converted from strings into SVM input format, in this case array of 0 and 1. PSSM profiles had to be transformed first in order to be used in following steps. It was done with np.genfromtxt function saving only frequency matrix as 2D array where each row was describing the probabilities of each amino acid in this position. The percentage values were stored as fractions to avoid biases. The lists were created in such way that same indexes in each list corresponded to the same protein.

* Creating sliding window and corresponding states.

In order to obtain input format accepted by SVM, array for each window were created. To avoid confusion, window length had to be odd number. The length of each array was as there are 20 numbers describing probabilities of amino acid in given position of sequence. For window length n the window in position(i) consisted of frequency arrays of residues from to . An important feature which had to be taken into consideration was solving border cases - windows which range was going over the ends of the sequence. In this cases instead of frequency information an array consisting of 20 zeros were added for each position over the range of the sequence. All windows for all proteins were stored together as 2D array with shape:

The corresponding states were appended in such way that the index of array of states was the same as index of window in all windows array.

* Cross-validation and model optimization

In order to obtain the generalized accuracy of the model, 3-fold cross validation was performed using cross\_val\_score function from sklearn library. 3-fold was chosen since it takes significantly less time to run compared to often used 10 fold cross validation. The parameters were tweaked one by one for window lengths between 3 and 31. All possible kernels for SVC(linear, polynomial, rbf and sigmoid) and also LinearSVC were tested. Cache\_size parameter was set to 3000 to speed up the process. Finaly, the results for two other methods – random forest classifier and simple decision tree were generated for same range of window length. Model was generated for best scoring SVC parameters using pickle and stored in results directory as PSSM\_model.

* Predictor programme and results generation

Program for prediction was written in similar way as modelling one. Provided fasta file with proteins of unknown exposure status, it generates windows, this time however instead of frequency matrix, sequence is converted into binary form. For each sequence in testing dataset, the exposure status is predicted based on previously generated model and stored in the results directory in the three line pattern. Results of all the optimizations were stored in MS excel, where later plots were generated. Receiver operating characteristic(ROC) curve and Matthews correlation coefficient (MCC) were generated using sklearn library functions.

## Results and discussion

In order to obtain best possible accuracy of model it is necessary to try different parameters of SVC. In this project, different kernels at different window lengths were tested first. The results are visible of *Figure 1.*

Figure Accuracy of SVC for different kernel types and window lengths

The accuracy values are presented as percentage values, which are the average of scores for 3 fold cross validation. The accuracy was much higher in case of both SVC with linear kernel and LinearSVC compared to other kernels. The highest accuracy was 74.73% and it was observed for SVC with linear kernel, window length of 17. The accuracy values were very similar for both polynomial and rbf kernel. In case of sigmoid kernel,

Task is to develop a SVM model and optimize it by in example checking the optimal window length and kernel and addition of more information i.e. evolutionary info to further increase its accuracy also to compare our SVC model with other stuff like random forest and simple decision tree to further asses its performance

Because they lack the pattern, not many proteins are known and no positive inside rule

- Exposed vs Buried

It is important for reasons stated in the article I previously read

- Some examples of best currently available with their accuracies

For SVM its around 75-80% - best approach seems to be HMM – paper references

Methods:

introduce SVM

And some python PSSM stuff how it works

Results:

Comparison of PSSM with no PSSM

Comparison different kernels

Comparison of different window lengths

Comparison of the SVM with random forest and single decision tree methods

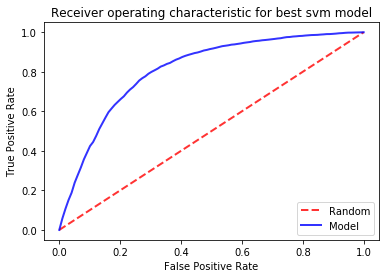
Things that did improved the linear SVM

Measurement of the Accuracy ROC curve MCC maybe

How our results correspond to the currently available predictor and also how SVM does in general compared to hmm

Conclusion:

What was achieved and how my predictor does what might be possible applications



## Conclusions