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

## Abstract

## 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. As far as SVM is concerned the best performance in exposure status found in literature is 77.91% for the membrane core regions and 80.42% in interface regions(Hayat, Park, & Helms, 2011). After further calculation based on data provided in the article, total prediction accuracy is 78,35%. There are some other approaches used to predict the exposure status, as stated by (Singh et al., 2011), the best available one is hidden markov model. TMBHMM which is such HMM exposure status predictor achieved prediction accuracy of 83%(Singh et al., 2011)

## 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 21. 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. Confusion matrix, 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 1 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. Linear kernel for SVC was the one which achieved the highest peak value of 74.73% and it was observed for window length of 17. Polynomial kernel characterized by degree = 4 and coefficient =2 obtained slightly lower results with higest score being 74.53% for the window length of 11. For polynomial kernel the accuracy scores were rising with higher degree of kernel, however this behaviour might be credited to overfitting and therefore I decided to pick degree of 4 as the highest value. LinearSVC was characterized by similar results as two abovementioned SVC kernels. Highest Accuracy value was obeserved for winowlength of 17 and it was 74.31%. Rbf kernel marked on the figure with dark blue color had significantly lover accuracy with highest value being 72.83% for window length of 15. In case of sigmoid kernel marked with yellow color, not all accuracy values were presented on the figure in the intrest of focusing on the more relevant results. In this case, the accuracy was rising and peaked 72.38% at window lenghth 5 after which it plummeted to the level of 51% for window length of 11, which is in fact almost random prediction, and remained on the level for the rest of tested window lengths. Since the accuracy of SVC with linear kernel was highest, it was taken for further comparison with other methods. Fore each kernel tested multiple additional tests were performed in order to provide the best possible parameters for model training. It was checked that changing of tolerance value didn’t improve the accuracy of models. When C value was manipulated, all changes from default value resulted in decreased scores. class\_weight parameter wasn’t changed since the dataset provided is rather balanced (6528 exposed residues and 6937 burried residues).

Figure 2 presents the accuracies of best svm model created with multiple sequence alignmnet input and svm model with highest score when input was a single sequence.

Figure 2 Comparison of the results with and without addition of evolutionary information

As clearly visible there is a noticeable improvement in prediction accuracy when evolutionary information was added. The difference between highest accuracies is more then 3% - 71.42% is highest for single sequence information(window length = 11) compared to 74.73% for highest accuracy when input was PSSM matrix. It is worth noticing that changes of tolerance in case of single sequence input resulted in slight improvement compared to default settings while similar changes in case of PSSM input did not result in any accuracy changes. <my teory – MSA already gives this sort of tolerance with different possible aa in one position>.

The results of comparison between SVC and two other cassification techniques, namely random forest classifier and decision tree classifier was presented in Figure 3. The highest accuracies for all window lenghts were achieved by random forest classifier, with higest accuracy 76.37% for window length equal 5. The results for SVC were already discussed above, as for other comparisons, Figure 3 Comparison of best SVC with decision tree and random forest classifier methods

SVC with best possible parameters was used. Decision tree clasifier predictions were significantly less accurate and peaked with 71.78% accuracy at windowlength of 11. For the highest Accuracy values achieved by each classifier, Matthews correlation coefficient was calculated, and it is presented in Table 1, supplied with optimal window length for each of classifiers.

Table 1 Matthews correlation coefficients for different classifiers

|  |  |  |
| --- | --- | --- |
| Classifier | Optimal window length | Matthews correlation coefficient (MCC) |
| SVC | 17 | 0,496201 |
| RFC | 5 | 0,532415 |
| DTC | 11 | 0,441449 |

Additional measurement were also performed for the model with optimal SVC parameters such as Receiver operating characteristic(ROC) curve presented on figure 4.

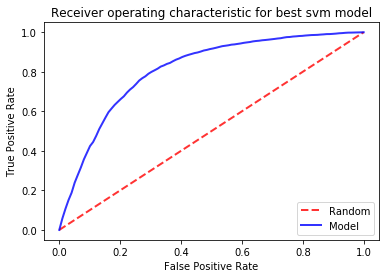


Figure 4 Receiver operating characteristic curve for optimal SVC parameters

## Conclusions

TODO:

* results ROC description
* finish confusion matrix – put into results
* write conclusions
* write abstract

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

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