EPiC Series in Computing
Volume 92, 2023, Pages 35–46
Proceedings of International Conference on Bioinformatics and Computational Biology (BICOB-2023)

Transmembrane Protein Inter-Helical Residue Contacts Prediction Using Transductive Support Vector Machines

Bander Almalki[1], Aman Sawhney[1], Li Liao [1]
[1] Department of Computer and Information Science, University of Delaware, DE 19716, USA.
alathwny@udel.edu, asawhney@udel.edu , liliao@udel.edu

Abstract

Protein functions are strongly related to their 3D structure. Therefore, it is crucial to identify their structure to understand how they function. Studies have shown that numerous numbers of proteins cross a biological membrane, called transmembrane (TM) proteins, and many of them adopt alpha helices shape. How these helices contact one another inside the membrane plays a major role in their tilt angle and relative position and hence the overall structure of the protein. To tackle the sparsity issue of labelled data, which is usually the case in amino acids residues contacts prediction, we adopt a transductive learning approach, which involves the unlabeled test data during training in order to obtain a better model. Using features extracted from protein structures, we compare transductive support vector machine (SVM) and inductive SVM in predicting helix-helix residues contacts to identify conditions and limitations where TSVMs gain performance and investigate the performance degradation of the TSVM and the best remedial solutions in the literature. In particular, we develop an early stop technique $TSVM_{ES}$ that generates a more accurate model and outperforms the state of art TSVM by 5%, as tested on a benchmark set of transmembrane proteins.

1 Introduction

Proteins are known to be the workhorse of life and one of the main focuses of bioinformatics research. They are responsible for many cell activities such as catalyzing various biochemical reactions, fighting infections by forming antibodies, providing cell and tissue structure, and transporting nutrients throughout the body [1]. A specific type of Membrane protein, called Transmembrane protein (TM), spans the entire cell membrane [2] and represents around a third of all living cell proteins. Even though its abundance and importance, only a limited number of TM protein structures have been determined experimentally [3]. In general, determining protein structure experimentally is known to be an expensive and time-consuming task [1]. Therefore, computational methods play an integral role in solving this dilemma [4]. Studies have shown that more than 36% of these TM proteins adopt alpha helix shape and can cross the membrane in a single-span (bitopic) or multi-span (polytopic) [5]. These helices consist of small amino acid residues that can be in contact with each other to determine the overall structure of the TM protein. The accurate prediction of these contact points is a significant intermediate step to building the overall 3D structure of the protein [6]. Success in predicting the ultimate 3D structure of a protein can reveal invaluable information about its functions and help to predict its behavior. Computational methods based on machine learning rely on labeled data to train models so that they can be used for predicting unlabeled test data and eventually for de novo prediction. Due to the aforementioned cost and other technical limitations, there is a sparsity issue of labeled data for helix-helix contact in transmembrane proteins. In this paper, unlike the current contacts prediction computational methods that use inductive learning approaches to predict TM protein inter-helical residues contact, we adopt a transductive learning approach. The idea of transductive learning is to allow the unlabeled test examples to participate in model training. This can be particularly useful when the test set is much bigger than the training set, which is usually the case in amino acids residues contacts prediction. Using features extracted from transmembrane protein structures to train classifier to predict helix-helix residues contact, we compare transductive SVM and inductive SVM to identify conditions and limitations where TSVM outperforms inductive methods. In particular, we investigate the performance degradation of TSVM and the currently best remedial solutions in the literature and develop an early stop technique $TSVM_{ES}$ that can generate a model to avoid performance degradation. Tested on a benchmark dataset of transmembrane proteins, our method significantly outperforms the state of art TSVM.

2 Methodology

For In the interest of clarity, we first provide a brief review of the used learning approaches, including the transductive learning models, and then discuss our proposed method and compare it with current TSVM models.

2.1 Inductive Learning vs Transductive Learning

Inductive learning refers to a process where the learner discovers rules by observing a set of examples N [7]. Then, the learner utilizes the training examples (and their labels),

$$(\vec{x}_1, y_1), (\vec{x}_2, y_2), \dots, (\vec{x}_n, y_n)$$

to build a model that fits the data. Then uses the model to predict the labels of unknown test examples (deduction). Examples of inductive learning in machine learning include traditional machine learning models, such as linear regression and support vector machines. The main disadvantage of inductive learning is the inability to generalize well when not seeing enough training examples. In contrast to inductive learning, a transductive learner has observed all the data beforehand, both the training set and testing set (without label) [8]. Therefore, the learner takes.

$$(\vec{x}_1, y_1), (\vec{x}_2, y_2), \dots, (\vec{x}_n, y_n) \; and \; \vec{x}_1^*, \vec{x}_2^*, \dots, \vec{x}_k^*$$

as input. Then, a function $f(x)$ is selected such that the expected prediction errors on the test examples are minimized [8]. Transductive learning can be very useful when the training set is very small, much smaller than the test data. Therefore, the transductive learner can utilize the test examples during training. However, the main assumption of transductive learning is that the data would form clusters in the features space which might not be always the case [8].

2.2 Transductive Support Vector Machines (TSVM) and Limitations

The support Vector Machines (SVM) algorithm was proposed in 1995 [9]. The main goal of SVM is to find a hyperplane that can distinctly separate data points of different classes. If the data can be linearly separable, SVM chooses a hyperplane that has the maximum margin $\frac{2}{||w||}$, where w is a vector normal to the hyperplane. However, since the data usually can't be linearly separable, the so-called soft margin SVM allows for misclassification, and the optimization function becomes,

$$\min(\frac{||w||}{2} + c \sum_{i=1}^{n} \xi_i)$$

where c is the misclassification penalty and ξ , called slack variable, measures how far the misclassified point is from its corresponding margin.

Transductive Support Vector Machines (TSVM) algorithm was introduced later to allow for the unlabeled test data to participate in training [10]. The algorithm tries to minimize misclassifications by utilizing test examples during training [Figure 1]. Then, the goal of transductive learner L is to find a hypothesis from the hypothesis space H, using training examples and test examples (without their labels) such that the misclassification is reduced.

$$R(L) = \int \frac{1}{k} \sum_{i=1}^{k} \theta(h_L(\vec{x}_i^*), y_i^*) dP\ (\vec{x}_1, y_1) \ldots dP(\vec{x}_k^*, y_k^*)$$

Where k is the number of test examples and h_L is the Hinge Loss. Then, the TSVM objective function is to minimize the following function.

$$\frac{1}{2}||\vec{w}||^2 + C\sum_{i=1}^{n} \xi_i + C_-^* \sum_{j:y_j^*=-1} \xi_j^* + C_+^* \sum_{j:y_j^*=1} \xi_j^*$$

Where $C, C_+^*,\ and\ C_-^*$ are hyperparameters for misclassification penalty of labeled examples, predicted positive unlabeled, and predicted negative unlabeled examples respectively.

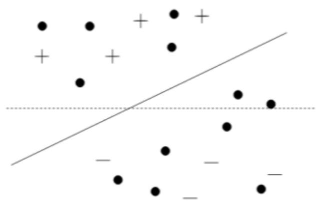

Figure 1 Inductive SVM (dotted line) VS Transductive SVM (solid line) margin

An early and popular implementation of TSVM, SVM_{light}, has been shown to improve the performance over inductive SVM in some cases [10]. However, it has been criticized for its unstable performance. For example, TSVM performs well on text classification but can perform substantially worse than SVM in other applications [11]. It has also been argued that there is a lack of evidence that the notion of separation leads to correct classification [12]. In addition, there have been suggestions that while the cost function of TSVM is appropriate, the implementation of TSVM may be inadequate [13].

It's worth noting that the TSVM optimization function is non-convex and there might be more than one local optimum. Therefore, many efforts have been made to solve this problem. In [14] an algorithm (TSVM-SA) is proposed to combine TSVM with simulated annealing (SA), which is an optimization technique to avoid being stuck in the local optimum, helping to find the global optimum. TSVM-SA overcomes the shortages of the original TSVM approach of having to estimate the ratio of positive/negative samples and achieved better generalization performance. However, it suffers from some major drawbacks, including the difficulty of fine-tuning to specific problems and long execution time, which might become unacceptable when the data size becomes large.

One of the major critiques of SVM_{light} [10] is its performance degradation which sometimes can be worse than the inductive SVM. Therefore, a safe Transductive Support Vector Machines, S4VM, is proposed [15], here "Safe" means that TSVM is never significantly worse than the inductive SVM. It is shown that SVM_{light} finds only one low-density separator with the maximum margin while there might be more than one. Hence, choosing the wrong separator might cause performance degradation. Therefore, S4VM finds all candidate separators using global simulated annealing and then chooses the best candidate using a local search mechanism [15]. The major drawback of S4VM is that it is extremely slow compared to SVM_{light}. As a result, S4VM is not recommended to be used with large datasets [15].

2.3 TSVM with early stop ($TSVM_{ES}$)

For the reasons mentioned above, we set to investigate how the different variants of TSVM will perform in predicting the helix-helix residue contacts, in comparison to the inductive SVM, under various settings as the test size increases. The results, as shown in the next section, confirms the "safe" property of S4VM, whereas reveals the performance degradation of the original TSVM implementation SVM_{light}. However, as we examine the data clustering property and model uncertainty, which are the possible reasons of performance degradation as suggested in the literature [17], no clear indication is found, as shown in the results section. Instead, by monitoring the accuracy and margin size of TSVM over each training iteration, we notice that the performance starts to drop when the margin size begins to decrease [Figure 3,4]. It turns out that, in SVM_{light}, the algorithm [Algorithm 1] loop 2 checks only for three conditions before applying label switching.

- Condition 1: the two points must have different labels (one positive and the other negative, or vice versa)
- Condition 2: the two points must be misclassified. Otherwise, there is no label switching.
- Condition 3: the distance from each point to the margin must be larger than 2.

What is missing, however, is that the greedy algorithm does not check whether the label-switching process would decrease the optimization function or not, possibly as a way to explore broader configurations, in the spirit of simulated annealing that those later algorithms such as S4VM have adopted.

```
Algorithm 1:
while $((C_-^* < C^*) \parallel (C_+^* < C^*))$ {                    // Loop1
    $(\vec{w}, b, \vec{\xi}, \vec{\xi}^*) := solve\_svm\_qp\ ([(\vec{x}_1, y_1) \ldots .(\vec{x}_n, y_n)], [(\vec{x}_1^*, y_1^*) \ldots .(\vec{x}_k^*, y_k^*)], C, C_-^*, C_+^*)$ ;
    $wihle\ (\exists m, l : (y_m^* * y_i^* < 0)\ \&\ (\xi_m^* > 0) \&\ (\xi_i^* > 0)\ \&\ (\xi_m^* + \xi_i^* > 2))$ {        // Loop 2
            $y_m^* := -y_m^*$ ;    // take +ve and −ve examples and switch their labels
            $y_i^* := -y_i^*$ ;
            $(\vec{w}, b, \vec{\xi}, \vec{\xi}^*) := solve\_svm\_qp\ ([(\vec{x}_1, y_1) \ldots .(\vec{x}_n, y_n)], [(\vec{x}_1^*, y_1^*) \ldots .(\vec{x}_k^*, y_k^*)], C, C_-^*, C_+^*)$ ;
            }
    }
```

Therefore, we propose $TSVM_{ES}$ a simple yet effective fix [Algorithm2] that can not only run faster than the current methods but also outperform the state of art TSVM (S4VM). This is done by monitoring the margin size in each training iteration using a separate validation set, after the label switching phase (loop 2), and enforcing an early stop technique to inhibit over-training. The modified algorithm does not use simulated annealing techniques which are expensive and can be sensitive to the annealing schemes. In the result section, we show that the new algorithm actually outperforms S4VM, the state-of-the-art TSVM.

```
Algorithm 2:
while $((C_-^* < C^*) \parallel (C_+^* < C^*))$ {                    // Loop1
    $(\vec{w}, b, \vec{\xi}, \vec{\xi}^*) := solve\_svm\_qp\ ([(\vec{x}_1, y_1) \ldots .(\vec{x}_n, y_n)], [(\vec{x}_1^*, y_1^*) \ldots .(\vec{x}_k^*, y_k^*)], C, C_-^*, C_+^*)$ ;
    $wihle\ (\exists m, l : (y_m^* * y_i^* < 0)\ \&\ (\xi_m^* > 0) \&\ (\xi_i^* > 0)\ \&\ (\xi_m^* + \xi_i^* > 2))$ {        // Loop 2
            $y_m^* := -y_m^*$ ;   // take +ve and −ve examples and switch their labels
            $y_i^* := -y_i^*$ ;
            $(\vec{w}, b, \vec{\xi}, \vec{\xi}^*) := solve\_svm\_qp\ ([(\vec{x}_1, y_1) \ldots .(\vec{x}_n, y_n)], [(\vec{x}_1^*, y_1^*) \ldots .(\vec{x}_k^*, y_k^*)], C, C_-^*, C_+^*)$ ;
            $\Gamma = [\,]$         // a list to store margin's size at eatch iteration
            $\Gamma$.append ( $ɱ := \frac{2}{||w||}$)                    //margin size
            $if\ \Gamma[-1] < \Gamma[-2]$ {
                break                // break Loop 2
            }
    }
```

3 Data and Results

3.1 Data

The dataset is adopted from [16] and consists of 222 α-helical TM proteins with a resolution better than 3.5 Å and with number of TM helices ranging from 2 to 17. The original dataset is divided into two sets,

I. training dataset, containing 165 chains (TRAIN), and

II. Testing dataset (TEST), containing 57 chains.

In this work, to reduce the running time, we use the TRAIN set only. When two helices are considered for inter-helical contact, they are often visualized as a two-dimension matrix (or map), with column corresponding to one helix and row corresponding to the other. So, for a residue pair at the position (i, j) in the contact matrix, we like to predict if it is a contact or non-contact by learning from features extracted from its neighboring residue pairs. The following neighboring positions are used: $(i, j)(i + x, j + y) where\ (x, y) = \{(1,1), (-1, -1), (1, -1), (-1,1), (0,1), (0, -1), (1,0), (-1,0)\}$. We have 40 features representing residue pair structural features including mean distance, standard distance, alpha-carbon distance, relative-residue angle, and inter-helical angle. For each feature set, the label 1 is given if there is a contact and zero otherwise. For simplicity, we balanced the dataset by down-sampling the number of contacts and non-contact examples to be equal.

3.2 Implementation and Results

As explained in the Introduction section, we tackle the sparsity issue of labeled data in predicting transmembrane protein inter-helical residue contact by adopting a transductive learning approach to incorporate the unlabeled test data into training. Specifically, we (1) apply a transductive SVM on a set of features extracted from transmembrane proteins structure as shown above to predict helix-helix residues contacts, and compare to the performance of the inductive SVM, (2) examine conditions and limitations where TSVM outperforms inductive SVM, (3) investigate the performance degradation of traditional TSVM (SVM_{light}). Moreover, (4) test our new method of an early stop technique $TSVM_{ES}$ to fix the performance degradation issue. The results show that it can outperform the state of art TSVM ($S4VM$) and produce a more accurate prediction with less computational time.

(1) Predicting TM Proteins inter-helical residue Contacts using TSVM, and comparing transductive and inductive approaches [Table 1]

Training examples	**Testing examples**	**Inductive SVM Accuracy**	**Inductive SVM F1**	**TSVM (SVM_{light}) Accuracy**	**TSVM (SVM_{light}) F1**	**Regularization**
1000	2000	0.8155	0.8442	0.835	0.8582	kernel=linear C=1, Cu=0.5
1000	4000	0.81025	0.84051	0.8595	0.8767	kernel=linear C=1, Cu=0.5
1000	4000	0.809	0.83963	0.8575	0.874945	shuffled kernel=linear C=1, Cu=0.5
1000	6000	0.8106	0.8408	0.802	0.8139	kernel=linear C=1, Cu=0.5
	4000+2000(1*)					
	4000+2000(2*)	0.8118	0.8416	0.7896	0.800	
1000	10000	0.8136	0.8407	0.8025	0.8097	kernel=linear C=1, Cu=0.5

Table 1 Inductive SVM VS Transductive SVM (SVM_{light}) performance. C and Cu are the labeled and unlabeled examples' penalties respectively.

(2) Identifying situations where TSVM outperforms inductive SVM. Table 1 shows that SVM_{light} outperforms inductive SVM, in the first three cases, in terms of accuracy and F1. Even though the learning performance of SVM_{light} is expected to be better than inductive SVM by exploiting more unlabeled test examples, surprisingly, the performance starts to degenerate as the number of test examples exceeds 4000. Here, the 6000-test set is the previous 4000-test set plus 2000 newly added examples. To exclude the possibility that the performance decrease is due to the peculiarity of these newly added 2000 examples, we randomly select two different 2000 example sets (a) and (b). Results suggest that it is the increase of test data size, instead of any specific examples, that causes the performance drop.

(3) Investigating the performance degradation of the traditional TSVM (SVM_{light}). To this end, we follow what are suggested in the literation as reasons linked to the TSVM performance drop [17]. These reasons are: (a) Data quality, where it assumes that data have inherent cluster structure and instances falling into the same cluster have the same class label. (b) Model uncertainty, in terms of TSVM, [15] claims that the model uncertainty is a result of the existence of multiple low-density separators, and choosing the incorrect one might cause performance degradation. (c) Measurements diversity, which claims that different tasks use different performance measurements and choosing the wrong metric can cause unstable performance. Failing to meet these three conditions can cause a significant drop in TSVM performance. In our situation, since our data set is balanced, accuracy and F1 metrics are used. Therefore, we can ignore (c) as a cause of the performance drop and focus on (a) and (b).

(a) To measure data quality and examine the clustering behavior of the 4000 and 6000 test sets, we conduct three experiments. First, we use the top two Principal Component Analyses (PCA) based on variance to reduce the data dimensionality to two dimensions to be able to visualize the two clusters using the K-means clustering algorithm. Figure 2 shows that 4000 and 6000 examples make very similar clusters. Second, we randomly sample different examples from the data set to exclude that the performance decrease is a result of the newly added 2000 examples. Table 1 shows that random sampling of the 4000-test set and the added 2000 examples in the 6000 set (1^*) and (2^*) does not affect the performance. Third, we use Inertia and Davis Bouldin score (DBS) metrics to measure the clustering quality [Table 2]. Inertia (Equation 1) is a metric used to calculate the sum of squared distances of samples to their closest cluster center while DBS (Equation 2) calculates the ratio of within-cluster distances to between-cluster distances. Results show that both 4000 and 6000 test sets have very similar clustering scores. Therefore, data quality can be excluded as a reason for TSVM performance degradation.

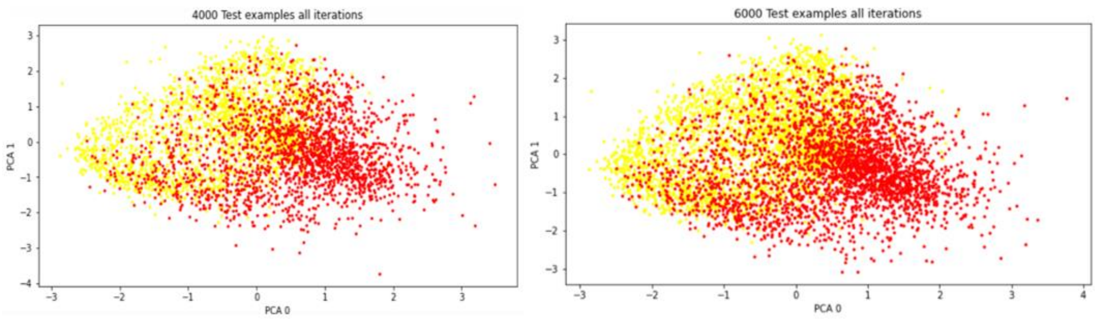

Figure 2 4000 test examples VS 6000 test examples clusters

$$J = \sum_{j=1}^{k} \sum_{i=i}^{n} \left\| x_i^{(j)} - c_j \right\|^2$$

Equation 1 Inertia Metric where k is the number of clusters, n the number of examples, c the center of the cluster.

$$DB = \frac{1}{k} \cdot \sum_{i=1}^{k} R_i \quad where$$

$$R_i = \max_{j=1,\dots k, i \neq j} R_{ij} \quad and \quad \mathrm{R}_{\substack{\mathrm{ij} \\ i \neq j}} = \frac{var(C_i) + var(C_j)}{\left\| c_i - c_j \right\|}$$

Equation 2 Davies Bouldin score where C_i and c_i refers to the i^{th}cluster, and its centroid respectively.

	4000 examples inertia/# of points	6000 examples inertia/# of points	4000 examples Davies Bouldin score	6000 examples Davies Bouldin score
K-means k=2	8.849	8.774	2.77	2.78

Table 2 4000 test examples VS 6000 test examples inertia and Davies Bouldin score. k is the number of clusters.

(b) To measure model uncertainty, we use S4VM model [15] on the same dataset above. S4VM claims to find all candidate SVM separators using global simulated annealing and then chooses the best candidate using a local search mechanism. Table 3 shows the S4VM performance, in terms of accuracy and F1, compared to the inductive SVM. It can be seen that S4VM outperforms inductive SVM in all settings and never degenerates significantly when increasing the size of the test set. Therefore, S4VM can be a solution for the problem SVM_{light} performance decrease. However, S4VM suffers from major issues which makes it unfavorable. For example, S4VM uses Simulated Annealing which can be a very time-consuming task to find an optimal solution. Therefore, the running time might become unacceptable when the data size is large.

Training set size	Testing set size	SVM Accuracy	S4VM Accuracy	SVM F1	S4VM F1
1000	2000	0.8155	0.842	0.8442	0.863
1000	4000	0.81025	0.862	0.84051	0.8778
1000	6000	0.8106	0.8625	0.8408	0.8784
1000	10000	0.8136	0.8548	0.8407	0.8721

Table 3 Inductive SVM Vs Transductive SVM (S4VM)

3.3 Results of running the proposed method $TSVM_{ES}$

We propose $TSVM_{ES}$ that can solve the problem of TSVM performance degradation and reduce the running time significantly. This is done by monitoring the margin size in each training iteration, after the label switching phase (loop 2), and enforcing an early stop technique to inhibit over-training.

Therefore, the training stops when the margin size starts to decrease [Figure 5,6]. The performance of our proposed method compared to the current TSVM models is shown in table 4 while the running time compared to the S4VM is shown in figure 7.

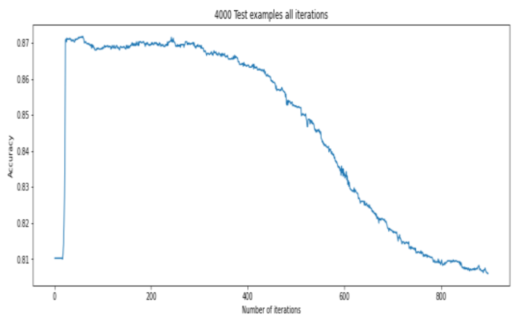

Figure 3 Transductive SVM (SVM_{light}) accuracy over training iterations

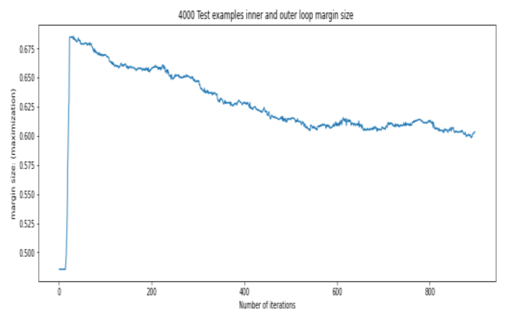

Figure 4 Transductive SVM (SVM_{light}) margin size over training iterations

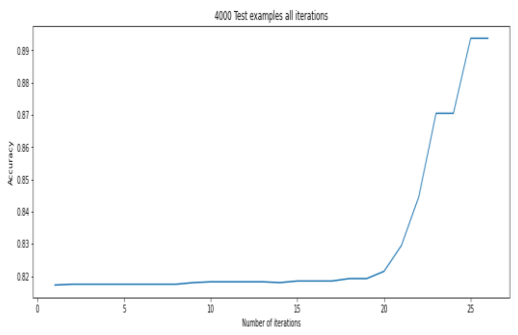

Figure 5 $TSVM_{ES}$ (Early stop) accuracy over training iteration

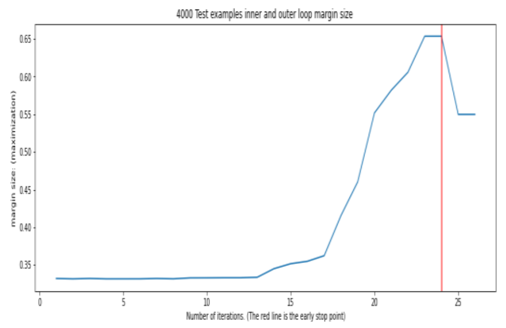

Figure 6 $TSVM_{ES}$ (Early stop) margin size over iteration

Training set	Test set	SVM_{Light} accuracy	S4VM accuracy	$TSVM_{ES}$ accuracy	SVM_{Light} F1	S4VM F1	$TSVM_{ES}$ F1
1000	2000	0.835	0.842	**0.8615**	0.858	0.863	**0.878**
1000	4000	0.859	0.862	**0.8875**	0.876	0.8778	**0.897**
1000	6000	0.802	0.8625	**0.8976**	0.813	0.8784	**0.905**
1000	10000	0.802	0.8548	**0.902**	0.809	0.8721	**0.907**

Table 4 TSVM (SVM_{Light}), TSVM (S4VM), and $TSVM_{ES}$ (Our method) performance

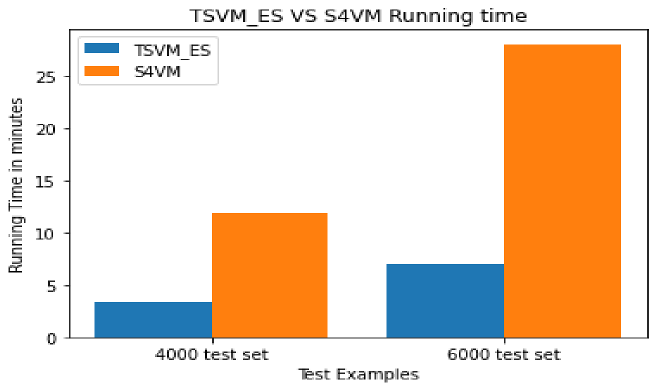

Figure 7 $TSVM_{ES}$ VS S4VM Running Time on two different test sets.

4 Conclusion

In this paper, to tackle the sparsity issue of labeled data in predicting transmembrane protein helix-helix residues contact, we adopted a transductive learning approach, which allows for the unlabeled test data to be used during training. We showed that Transductive Support Vector Machines can improve the classifier prediction performance, especially when the test set is bigger than the training set, which is usually the case in amino acids residues contacts prediction. We proposed a new method that uses an early stop technique by monitoring the TSVM margin to prevent over-training and solve the problem of the performance degradation of the traditional TSVM (SVM_{light}), and demonstrated that the proposed method is very effective and can outperform the state of art TSVM (S4VM) in achieving significantly better performance and reduce the running time significantly. While this method is proposed for the helix-helix residue contact prediction, it is reasonable to believe the technique can be applicable to other applications where TSVM is used to tackle the sparsity issue of labeled data.

References

[1] Breda A, Valadares NF, Norberto de Souza O, et al. (2006). Protein Structure, Modelling, and Applications. 2006 May 1 [Updated 2007 Sep 14]. In: Gruber A, Durham AM, Huynh C, et al., editors. Bioinformatics in Tropical Disease Research: A Practical and Case-Study Approach [Internet]. Bethesda (MD): National Center for Biotechnology Information (US); 2008. Chapter A06. Available from: https://www.ncbi.nlm.nih.gov/books/NBK6824/

[2] Mbaye, M.N., Hou, Q., Basu, S. et al. (2019). A comprehensive computational study of amino acid interactions in membrane proteins. Sci Rep 9, 12043 . https://doi.org/10.1038/s41598-019-48541-2

[3] Xia, Y., Fischer, A. W., Teixeira, P., Weiner, B., & Meiler, J. (2018). Integrated Structural Biology for α-Helical Membrane Protein Structure Determination. Structure, 26(4), 657-666.e2. https://doi.org/10.1016/j.str.2018.02.006

[4] Xiong, J. (2006). Essential Bioinformatics. Cambridge: Cambridge University Press. doi:10.1017/CBO9780511806087

[5] Goddard, A., Oates, J., Watts, A. (2013). Membrane Proteins: Structure and Organization. In: Roberts, G.C.K. (eds) Encyclopedia of Biophysics. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-16712-6_748

[6] Torrisi, M., Pollastri, G., & Le, Q. (2020). Deep learning methods in protein structure prediction. Computational and Structural Biotechnology Journal, 18, 1301–1310. https://doi.org/10.1016/j.csbj.2019.12.011

[7] Liang, T. P., Chandler, J. S., & Han, I. (1990). Integrating statistical and inductive learning methods for knowledge acquisition. *Expert Systems With Applications*, *1*(4), 391–401. https://doi.org/10.1016/0957-4174(90)90048-y

[8] Gammerman, A., Vovk, V., & Vapnik, V. (1998). Learning by transduction. Uncertainty in Artificial Intelligence, 148–155. https://arxiv.org/pdf/1301.7375.

[9] Cortes, C., Vapnik, V (1995). Support-vector networks. Mach Learn 20, 273–297. https://doi.org/10.1007/BF00994018

[10] Joachims, T. Transductive Inference for Text Classification using Support Vector Machines. Proceedings of the International Conference on Machine Learning (ICML), 1999. https://www.cs.cornell.edu/people/tj/publications/joachims_99c.pdf

[11] Wu, D., Bennett, K. P., Cristianini, N., & Shawe-Taylor, J. (1999). Large Margin Trees for Induction and Transduction. *International Conference on Machine Learning*, 474–483. https://dblp.uni-trier.de/db/conf/icml/icml1999.html#WuBCS99

[12] Zhang, T. and Oles, F. (2000) The value of unlabeled data for classification problems. In ICML, pp. 1191–1198, 2000.

[13] Chapelle, O. and Zien, A. (2005) Semi-supervised learning by low-density separation. In AISTATS, pp. 57–64, 2005.

[14] Sun, F., & Sun, M. (2005). Transductive Support Vector Machines Using Simulated Annealing. Computational Intelligence and Security, 536–543. https://doi.org/10.1007/11596448_78

[15] Li, Yu-Feng & Zhou, Zhi-Hua. (2010). S4VM: Safe Semi-Supervised Support Vector Machine. CoRR. abs/1005.1545.

[16] Sun, J., & Frishman, D. (2020). DeepHelicon: Accurate prediction of inter-helical residue contacts in transmembrane proteins by residual neural networks. Journal of Structural Biology, 212(1), 107574. https://doi.org/10.1016/j.jsb.2020.107574

[17] Li, Y. F., & Liang, D. M. (2019). Safe semi-supervised learning: a brief introduction. Frontiers of Computer Science, 13(4), 669–676. https://doi.org/10.1007/s11704-019-8452-2