Randomized Subspace Learning for Proline Cis-Trans Isomerization Prediction

Omar Y. Al-Jarrah, Paul D. Yoo, Kamal Taha, Sami Muhaidat, Abdallah Shami, and Nazar Zaki

Abstract—Proline residues are common source of kinetic complications during folding. The X-Pro peptide bond is the only peptide bond for which the stability of the cis and trans conformations is comparable. The cis-trans isomerization (CTI) of X-Pro peptide bonds is a widely recognized rate-limiting factor, which can not only induces additional slow phases in protein folding but also modifies the millisecond and sub-millisecond dynamics of the protein. An accurate computational prediction of proline CTI is of great importance for the understanding of protein folding, splicing, cell signaling, and transmembrane active transport in both the human body and animals. In our earlier work, we successfully developed a biophysically motivated proline CTI predictor utilizing a novel tree-based consensus model with a powerful metalearning technique and achieved 86.58 percent Q2 accuracy and 0.74 Mcc, which is a better result than the results (70-73 percent Q2 accuracies) reported in the literature on the well-referenced benchmark dataset. In this paper, we describe experiments with novel randomized subspace learning and bootstrap seeding techniques as an extension to our earlier work, the consensus models as well as entropy-based learning methods, to obtain better accuracy through a precise and robust learning scheme for proline CTI prediction.

Index Terms—Proline cis-trans isomerization, machine learning, subspace learning, ensemble methods

1 Introduction

The prediction of proline cis-trans-isomers of proteins based on amino acids remains a challenging problem in proteomics. The peptide bond between non-proline amino acids is much more stable in the trans than in the cis conformation. The large difference in the stability of the two isomers keeps the trans form 100 to 1,000 times more populated than the cis isomer [1], [2], [3]. Due to the high number of peptide bonds in the protein, the fraction of protein molecules that have cis peptide bonds in the denatured ensemble can be significant.

Since isomerization of peptide bonds is slow, non-native isomers of peptide bonds can also be slow in protein folding. It has been demonstrated that CTI of non-prolyl peptide bonds can give rise to significant slow folding phases [4]. The CTI of peptide bonds and the formation of disulfide bridges are slow steps that form bottlenecks in the proteinfolding reaction. Under certain circumstances, such processes can be linked and facilitate the formation of the correct disulfide bonds in the presence of peptidyl-prolyl cis-trans isomerase [5]. The importance of the cis-trans isomerization (CTI) as rate-determining steps in protein folding reactions has been well reported in the literature [6], [7], [8], [9], [10].

The first attempt to predict the CTI of proline using a computational model from amino acids was made by Froömmel and Preissner in 1990 [11]. They had taken adjacent/local residues (±6) of prolyl residues and their physicochemical properties into account, and found six different patterns that allow one to assign correctly about 72.7 percent (176 cis-prolyl residues in their relatively small dataset of 242 Xaa-Pro bonds of known cis-prolyl residues), where by no false positive one is predicted. Since Frömmel and Preissner's seminal work, various non-parametric machine learning models have been proposed. Recently, support vector machines (SVMs) seemed to be the most suitable for proline CTI prediction task. The first SVM-based computational predictor was built by Wang et al. [12]. They constructed a SVM with polynomial kernel function and used amino acid sequences as input, and achieved the Q2 accuracy of 76.6 percent. Song et al. [13] built a SVM with radial basis function, and used evolutionary information represented in position-specific-scoring matrix (PSSM) scores generated by PSI-BLAST [14] and predicted secondary structure information obtained from PSI-PRED [15] as input. Their SVM-based proline CTI predictor showed Q2 accuracy of 71.5 percent, and Mcc of 0.40. Pahlke et al. [16] demonstrated the importance of protein secondary structure information in the prediction of proline CTI residues. Their computational algorithm called COPS—the first attempt to predict for all 20 naturally occurring amino acids whether the peptide bond is a protein is in cis or trans conformation—used secondary structure information of amino acid triplets only. Most recently, Exar-chos et al. [17] developed a SVM with a wrapper feature selection algorithm on evolutionary information (i.e., PSSM scores), predicted secondary structure information, real-valued solvent, and accessibility level for each amino acid, and the physicochemical properties of the neighboring residues as input. They achieved 70 percent accuracy in the prediction of the peptide bond conformation between any two amino acids only. The recent computational proline CTI predictors have

- *O.Y. Al-Jarrah, K. Taha, and S. Muhaidat are with ECE Dept., Khalifa University, Abu Dhabi. E-mail: omar.aljarrah@kustar.ac.ae,.*
- *P.D. Yoo is with the Data Science Institute, Bournemouth University, United Kingdom. E-mail: paul.d.yoo@ieee.org.*
- *A. Shami is with the ECE Department, University of Western Ontario, London, Canada. E-mail: ashami@eng.uwo.ca.*
- *N. Zaki is with the College of Information Technology, UAE University, UAE. E-mail: muhaidat@ieee.org.*

Manuscript received 9 Oct. 2014; accepted 28 Oct. 2014. Date of publication 9 Nov. 2014; date of current version 4 Aug. 2015.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TCBB.2014.2369040

utilized machine-learning models such as SVM and its variants, with evolutionary (i.e., PSSM scores), and secondary structure information as input. Such models reached about 70-73 percent Q2 accuracies and 0.40 Mcc. This observation is aligned with the results of other computational biology studies [18], [19], [20], [21], [22]. Generally, SVM demonstrates its learning ability in the prediction/classification tasks in the fields of computational biology and bioinformatics [18], [19], [20], [21], [22].

In our earlier work, we introduced a novel approach that utilizes biophysically-motivated intelligent consensus model with a powerful randomized metalearning technique through the use of sequence information only (i.e., PSSMs generated by PSI-BLAST) for the prediction of proline CTI residues. The proposed model was built based on the idea of RandomForest data modeling [22], and evolutionary information. And it achieved 86.58 percent Q2 accuracy and 0.74 Mcc, which is a better result than the results (70-73 percent Q2 accuracies) reported in the literature on the well-referenced benchmark dataset [13]. Building upon our earlier results, we propose novel randomized subspace learning and bootstrap seeding techniques to obtain better accuracy through a precise and robust learning scheme for proline CTI prediction.

2 Methods

Our experiments consist of four consecutive phases. First, the collection of high-quality proline CTI proteins, and the pre-processing and construction of a proline CTI benchmark dataset. Second, the construction of each model and tune its parameters. In this phase, the proposed models and the a few selected models in the literature are constructed, and through a set of experiments, threshold values on the probability output of each classifier are chosen in order to optimize performance measure. The models we have chosen from the proline CTI prediction literature are SVM$_{\mathrm{LIB}}$ [23], SVM$_{\mathrm{ADA}}$ (Adaboost Lib-SVM) [24], Random Tree (RT) [25], Intelligent Voting Method (IVM) [26], and Randomized Metalearning Method (RMM) [26]. Third, the predictive performance of the proposed models is compared with the ones selected from the literature in terms of Accuracy (Acc), Sensitivity (Sn), Specificity (Sp), Mathew's correlation coefficient (Mcc), False Positive/Negative Rates (FPR/FNR), and Stability (Var) on the proline CTI dataset built in the first phase. We finally compare results with the consensus results from literature.

2.1 Evolutionary Dataset Construction

To make a fair comparison with existing proline CTI prediction models, we have chosen Song et al.'s [13] dataset. The dataset has 2,424 non-homologous protein chains, obtained from the Culled PDB list provided by PSICES server [27]. All the tertiary structures in the dataset were determined by X-ray crystallography method with resolution better than 2.0 Å and R-factor less than 0.25. In addition, the sequence identity of each pair of sequences is less than 25 percent, and the protein chains with sequence length shorter than 60 amino acids were excluded in the dataset. In total, there are 609,182 residues, and every sequence contains at least one *proline* residue. The PDB codes, CisPep PDB codes, proline *cis* peptide records, corresponding *dihedral* angles and protein sequences of the 2,424 protein chains used in this study are available on request.

Evolutionary information in the form of PSSM was included in the windows as direct input. Evolutionary information in form of PSSM is the most widely used input form for protein structure prediction in 1D, 2D and 3D, as well as other computational/structural proteomic prediction or classification tasks [15], [16], [17], [18], [19], [20], [21], [22]. The idea of using evolutionary information in the form of PSSM was first proposed by Jones [28], and generally, it has improved the accuracy about 3-5 percent in the prediction tasks.

To generate PSSM scores, we use the *nr* (non-redundant) database and *blastpgp* program obtained from NCBI [29]. We run *blastpgp* program to query each protein in our dataset against the *nr* database to generate the PSSMs with the following setup: 1) three iterations, 2) cutoff e-value of 0.001. Finally, the PSSM scores are scaled to the range between 0-1 by the following standard logistic function:

$$f(x) = \frac{1}{1+\exp(-x)},$$

where x is the raw profile matrix value. The scaled PSSM scores are used as direct input to the learning models. A PSSM is generated for each protein sequence, and has a $M \times 20$ matrix, where M is the target sequence length, and 20 is the number of amino acid types. Each element of the matrix represents the log-odds score of each amino acid at one position in the multiple alignments. The window size $2l + 1$ indicates the scope of the vicinity of the target prolyl peptide bonds, determining how much neighboring sequence information is included in the prediction. We select the windows size (l) of 9, and built our models as it produced the best predictive results, aligned with Song et al.'s experimental result.

When a large difference between positive and negative samples is observed in a training set, data imbalance problem exists [30]. Our dataset is composed of 1,265 *cis* and 27,196 *trans* residues. There are two general approaches to reduce such imbalance problem. First, increasing the number of under-samples by random resampling. Second, decreasing the number of over-samples by random removal. In this study, we adopt the first approach, and make 1 to 1 ratios between the sizes of positive (*cis*) and negative (*trans*) training samples.

2.2 Protein Secondary Structure Information

The recent computational proteomic studies report that protein secondary structure information is useful for various protein sequence-based classifications tasks [15], [16], [17], [18], [19], [20], [21], [22]. Although the mutations at sequence level can obscure the similarity between homologs, the secondary-structure patterns of the sequence remain conserved. That is because changes at the structural level are less tolerated. The recent studies mostly use the probability matrix of secondary structure states predicted from PSI-PRED [15]. PSI-PRED is a well-known computational predictor, and it predicts protein secondary structures in three different states (α-helix, β-sheet, and loop). However, there is one significant limitation with using predicted secondary structure

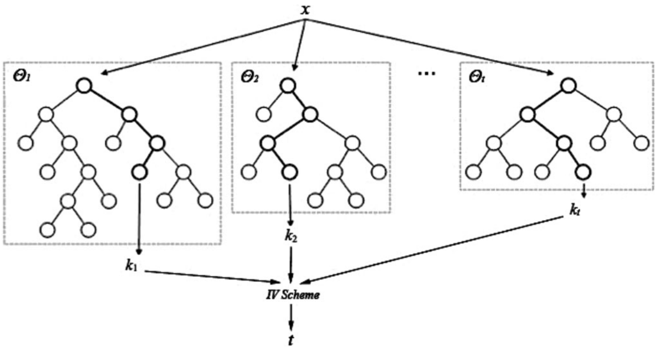


Fig. 1. A general architecture of method I Ensemble. The collection of randomized recursive decision trees $\{h(x, \Theta_k), k = 1 \ldots\}$, where the Θ_k are independently, identically distributed DRTs, and each RDT provides class probability of input *x*.

information. The best secondary-structure prediction model still cannot reach the upper boundary of its prediction accuracy. In other words, it is not good enough yet to be used as a confirmation tool. It shows about 75-80 percent Q3 accuracies only. Clearly, incorrectly predicted secondary structure information if presented in input dataset of a computational prediction/classification model leads to the poor learning and, eventually, to incorrect prediction of proline CTI residues. Although the predicted secondary information may be useful in some extent, it should not be used if one attempts to reach better than 80 percent Q2 accuracy. We, therefore, used evolutionary information in the form of PSSM obtained from protein amino-acid sequences only. To achieve above 80 percent Q2 accuracy, we believe that an accurate and correct information encoding presented in input dataset is critical, especially if used with intelligent/model-free modeling like machine-learning. In other words, noise presented in input dataset could lead to significant degradation in the performance of the models.

2.3 Method I: Randomized Subspace Learning

Recursive decision tree (RDT) is one of the most popular learning algorithms due to its simplicity, robustness, and ease of implementation [1]. It has first been introduced by J. Rose Quinlan as inductive logic programming method (ID3), and continually improved by machine-learning scientists [2].

In building such trees, the *gini* index is used as a decision function for determining the final class in each decision tree. The *gini* index of node impurity is the measure most commonly chosen for classification-type problems. If a dataset *T* contains examples from *n* classes, *gini* index *G(T)* is defined as:

$$G(T) = 1 - \sum_{j=1}^{n} (P_j)^2,$$

where p_j is the relative frequency of class *j* in *T*. If a data set *T* is split into two subsets T_1 and T_2 with sizes N_1 and N_2 respectively, the *gini* index of the split data contains examples from *n* classes, the *G(T)* is defined as:

$$G_{split}(T) = \frac{N_1}{N} G(T_1) + \frac{N_2}{N} G(T_2).$$

The attribute value that provides the smallest $G_{split}(T)$ is chosen to split the node. Since complicated classifiers tend to overfit the data and generalize poorly, the idea of combining multiple simple classifiers to form a strong one has been adopted. Such classifiers are called ensemble classifiers. The proposed randomized subspace partitioning ensemble classifier uses randomly selected C4.5 trees as base classifiers. Here, the algorithm builds the structure of N random C4.5 recursively. The feature set $X = \{F_1, \ldots F_K\}$ is used to construct the tree structure. x is finally classified by averaging the probability output from the N random C4.5 to estimate its posterior probability as follows:

```
classify ($\{T_1, \ldots, T_N\}, x$),
where T is eigenspace partitioned C4.8; and x is the new instance shall be labeled.
begine
        for each $T_i$
                $P_i(y/x) = \frac{n[y]}{\sum_y n[y]}$,
    where $n[y]$ is the count at the leaf
    that x finaally reaches to
return
        $\frac{1}{N}\sum_{i=1}^{N} P_i(y/x)$ for all class label y,
end
```

The algorithm grows multiple randomized trees by pseudorandomly selecting subsets of subspaced features of the feature vector. Each randomized tree uses K randomly selected features at each node and performs no pruning as depicted Fig. 1.

2.4 Method II: Bootstrap Seeding

Method I builds an ensemble of partitioned trees, and averages their classifications. Each one is based on the same input data; however, it uses a different random-number seed. Some learning algorithms already have a built-in random component. For example, when learning multiplayer perceptron using the back propagation algorithm, the initial network weights are set to randomly chosen values. One way to make the predictive performance of a learner stable is to run the learner several times with different random number seeds (i.e., initial weights) and combine the classifiers' predictions by voting or averaging. Learning in Method I builds a randomized partitioning tree in each iteration of the bagging algorithm, and often produces excellent predictors. Although bagging and randomization may yield similar results, it is worth to pay attention to such initial weighting method as they introduce randomness differently and may affect the performance of the classifier as a whole.

In Method II, the level of randomness is increased by introducing a bootstrap seeding technique. To decrease the correlation between the different classifiers, each classifier is trained on a bootstrap sample of the training data. First, a random seed is chosen which pulls out a random collection of samples from the training dataset while maintaining the class distribution. Second, with this selected dataset, a random subspace ensemble classifier is build. Where *F* is the total number of input attributes in the dataset, only *K* attributes are chosen randomly for each recursive tree, where $K < F$. To classify a new input vector X, the input vector

TABLE 1
Model Performance Comparisons in Different Fold

Models	Fold	Acc	Sp	Sn	Mcc
SVM$_{LIB}$	7	0.7613	0.5604	0.9621	0.5712
	8	0.7633	0.5623	0.9645	0.5756
	9	**0.7672**	**0.5722**	**0.9622**	**0.5813**
	10	0.7606	0.5584	0.9628	0.5702
SVM$_{ADA}$	7	0.7645	0.5647	0.9643	0.5647
	8	0.7647	0.5644	0.9650	0.5781
	9	**0.7653**	**0.5656**	**0.9650**	**0.5796**
	10	0.7564	0.5483	0.9645	0.5205
RT	7	0.8241	0.6956	0.9522	0.6724
	8	0.8278	0.6900	0.9656	0.6825
	9	0.8203	0.6917	0.9217	0.6360
	10	**0.8331**	**0.7067**	**0.9594**	**0.6890**
IVM	7	0.8032	0.6510	0.9554	0.6257
	8	0.8080	0.6644	0.9514	0.6444
	9	0.8077	0.6611	0.9544	0.6461
	10	**0.8150**	**0.6698**	**0.9599**	**0.6595**
RMM	7	0.8452	0.7399	0.9504	0.7077
	8	0.8575	0.7643	0.9506	0.7290
	9	**0.8658**	**0.7816**	**0.9500**	**0.7443**
	10	0.8589	0.7688	0.9489	0.7311
Method I	**7**	**0.9633**	**0.9889**	**0.9377**	**0.9284**
	8	0.9606	0.9833	0.9378	0.9232
	9	0.9570	0.9717	0.9389	0.9127
	10	0.9600	0.9833	0.9361	0.9227
Method II	7	0.8850	0.9269	0.9233	0.7864
	8	0.9364	0.9261	0.9217	0.8743
	9	0.9331	0.9372	0.9311	0.8698
	10	**0.9431**	**0.9678**	**0.9183**	**0.8877**

The parameters of each model were given the following values: ***SVM$_{LIB}$*** *(SVMType: C-SVM, cacheSize: 40.0, coef0: 0.0, cost: 13, debug: false, degree: 3, eps: 0.0010, gamma: 0.0, kernelType: rbf, loss: 0.1, normalize: false, nu: 0.5, seed: 1, shrinking: true), and* ***SVM$_{ADA}$*** *(the same as SVM$_{LIB}$'s, and for Adaboost, numIterations: 14, seed: 1, weightThreshold: 100),* ***RT*** *(KValue:0, minNum:1.0, minVarianceProp:0.001,Seed:1),* ***IVM*** *(debug: false, maxDepth: 0, numExecutionSlots: 1, numFeatures: 0, numTrees: 13, printTrees: false, seed: 1),* ***RMM*** *(the same setting for IVM, and for RMM, seed: 3 and iteration: 10),* ***Method I*** *(numExecutionSlots:1, numIterations:10, Seed:1, subSpaceSize:0.5),* ***Method II*** *(bagSizePercent:100, numExecutionSlots:1, numIterations:10, Seed:1).*

TABLE 2
FPR and FNR in Different Fold

Models	Fold	FPR	FNR	Var.
SVM$_{LIB}$	7	0.4396	0.0379	2.6810
	8	0.4378	0.0355	2.2972
	9	**0.4278**	**0.0378**	**2.5355**
	10	0.4416	0.0372	3.1828
SVM$_{ADA}$	7	0.4353	0.0357	2.1896
	8	0.4356	0.0350	2.3260
	9	**0.4344**	**0.0350**	**2.2619**
	10	0.4517	0.0355	3.5211
RT	7	0.3041	0.0478	1.9547
	8	0.3100	0.0344	3.006
	9	0.3083	0.0783	3.0516
	10	**0.2933**	**0.0406**	**3.0442**
IVM	7	0.3490	0.0446	1.0503
	8	0.3356	0.0486	1.3951
	9	0.3389	0.0456	1.2019
	10	**0.3302**	**0.0401**	**1.8248**
RMM	7	0.2601	0.0496	1.1392
	8	0.2357	0.0494	1.5232
	9	**0.2183**	**0.0500**	**2.2673**
	10	0.2312	0.0511	2.0655
Method I	**7**	**0.0111**	**0.0623**	**2.6921**
	8	0.0167	0.0622	2.3392
	9	0.0283	0.0611	2.8103
	10	0.0167	0.0639	2.1803
Method II	7	0.1534	0.0767	6.8595
	8	0.0489	0.0783	2.3407
	9	0.0628	0.0689	3.2596
	10	**0.0322**	**0.0817**	**2.7538**

FPR means experimentally verified trans residues that are predicted (incorrectly) to be cis residues; FNR indicates experimentally verified cis residues that are predicted (incorrectly) to be trans residues.

X is fed to each of the classifiers, then the ensemble algorithm averages the outputs from each classifier.

To estimate the performance of the ensemble algorithm, Method II performs a kind of cross-validation by using out-of-bag (OOB) data. Since each classifier in the ensemble grows on a bootstrap sample of the data, the sequences left out of the bootstrap sample, i.e., the OOB data, can be used as legitimate test set for that classifier. On average, $1 - e^{-1} \cong 1/3$ of the training data will be OOB for a given classifier. Consequently, each PSSM in the training dataset will be left out of $1/3$ of the classifiers in the ensemble, and use these OOB predictions to estimate the error rate of the full ensemble.

2.5 Model Validation and Testing

For the system model to be useful, it must be validated to ensure that it emulates an actual system in the desired manner. This is especially true for empirical models, such as statistical machine-learning models, which primarily rely on observed data. The validation of these models using problem-specific information, such as theoretic relationships or experimental knowledge, should be performed. There are several methods to perform the validation task, including, but not limited to re-substitution, cross-validation, bootstrapping, and their variants.

To accurately assess the predictive performance of each model, we adopt a cross-validation scheme for our model evaluation. First, we apply the holdout method to our proline CTI dataset. However, the holdout method has a key drawback in that the single random division of a sample into training and testing sets may introduce bias in model selection and evaluation. Since the estimated classification rate can be very different depending on the characteristic of the data, the holdout estimate can be misleading if we happen to get an unfortunate split. Hence, in our experiment, we adopt multiple train-and-test experiments to overcome the limitation of the holdout method. We create seven to 10-fold dataset, and only one of each fold is used for testing. The result of each fold is provided in Tables 1 and 2.

2.6 Parameter Tuning

All of the stages contain parameters or variables that need to be given appropriate values. Some of these parameters are so delicate that they have to be selected by an expert in the field, and kept constant thereafter. However, profoundly more interesting are the parameters the system is able to learn autonomously from training with available data. In this work, the accuracy of a classifier is optimized by choosing a mid-point threshold point on the probability output of the classifier. The threshold point is set using the *Weka ThresholdSelector* meta-classifier [31]. *Weka ThresholdSelector* allows the optimization of different performance measures on the 7-10 folded datasets using cross-validation.

3 Model Evaluation and Analysis

The performance of each model used in this study is measured by the Q2 accuracy (Acc: the proportion of true-positive and true-negative residues with respect to the total positives and negatives residues), the sensitivity (Sn: also called *recall*, the proportion of correctly predicted isomerization residues with respect to the total positively identified residues), the specificity (Sp: also called *precision*, the proportion of incorrectly predicted isomerization residues with respect to the total number of proline isomerization residues), and Mathew's correlation coefficient (Mcc: a correlation coefficient between the observed and predicted binary classifications, between -1 and $+1$). In Mcc, a coefficient of $+1$ represents a perfect prediction, 0 no better than random prediction and -1 indicates total disagreement between prediction and observation. Hence, a higher value of Mcc means that the model is more robust. The above measures can be obtained using the following equations:

$$Q2 = \frac{TP + TN}{TP + TN + FP + FN},$$
$$Sp = \frac{TN}{TN + FP},$$
$$Sn = \frac{TP}{TP + FN},$$
$$Mcc = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}},$$

where TP is the number of true postives, FN is the number of false negatives or under-predictions, TN is the number of true negatives, and FP is the number of false positives or over-predictions. We adopt the polynomial kernel function and radial basis function (rbf kernel) to construct the SVM classifiers, which is aligned with the existing proline CTI prediction studies [13]

$$K(\overrightarrow{x}_i \cdot \overrightarrow{x}_j + 1)^d,$$
$$K(\overrightarrow{x}_i \cdot \overrightarrow{x}_j) = \exp(-r\|\overrightarrow{x}_i - \overrightarrow{x}_j\|^2),$$

where the degree d needs to be tuned as for polynomial function, and the gamma and the regulator parameters for RBF need to be regulated. See the footnote of Table 1 for the parameters settings used for this study. For the optimal learning of the prediction models, the most suitable data fold for each model should be sought.

Table 1 shows the comparisons of the proposed models of this study, Methods I and II, with the original model, Random Tree, the popular SVM$_{\text{LIB}}$ and its variant, SVM$_{\text{ADA}}$, and the models proposed in the most recent study [26], Intelligent Voting Method and Randomized MetaLearning Method. The best score in each category is underlined, and the best fold scores in each model are bolded.

As in Table 1, the models proposed in our previous study [26], IVM and RMM, outperformed all SVM-type models. The proposed methods (Methods I and II) performed a far better than those of IVM and RMM, and all SVM-types achieving 96.33, and 94.31 percent Q2 accuracies respectively.

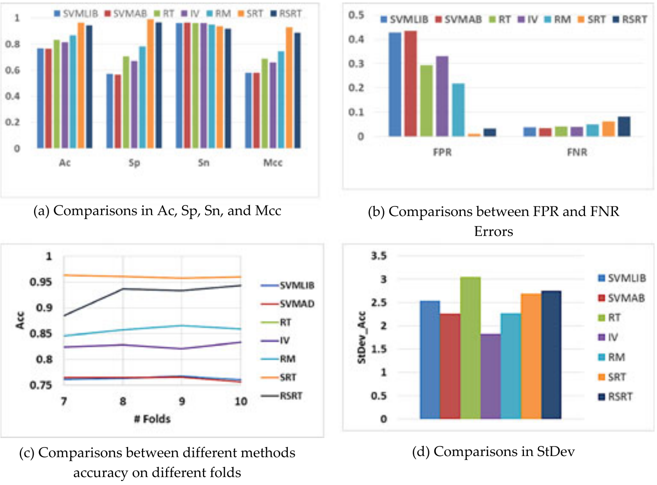
Both Methods I and II increased the values of Sp and Mcc significantly. Method I showed Sp of 98.89 percent and Mcc of 92.84 percent, meaning that the model is very robust. Randomized subspace learning technique used in Method I is proven to be useful as it shows Q2 accuracy far better than its original RT model. Interestingly, Methods I and II show higher Sp values while all other models show higher Sn values. Our experimental results shown in Table 1 demonstrates the success of randomized subspace learning and bootstrap seeding techniques for proline CTI prediction in Song's benchmark dataset. We believe this is the first time accurate computational prediction has been reported achieving Q2 accuracy above 90 percent.

As seen in Table 2, the performance of each model is also measured by false positive rate (FPR) and false negative rate (FNR). FPR means experimentally verified trans residues that are predicted (incorrectly) to be cis residues while FNR indicates experimentally verified cis residues that are predicted (incorrectly) to be trans residues. Generally, we observe not much difference in FNR; however, there is a significant improvement in FPR with Methods I and II. It means that Methods I and II have significantly reduced the misprediction rate predicting experimentally verified cis residues to be trans residues. SVM$_{\text{ADA}}$ shows the lowest FNR of 0.0350 while Method I obtains the lowest FPR of 0.0111.

Model variance (Var.) provides a good idea on model stableness and generalization ability. Although nonparametric machine-learning models have proved to be useful in many different applications, their generalization capacity has often been questioned because of the potential for model overfitting. The symptom of overfitting is that the model fits the training sample too well, and thus the model output becomes unstable for prediction. On the other hand, a more stable model, such as a linear model, may not learn enough about the underlying relationship, resulting in underfitting the data. It is clear that both underfitting and overfitting will affect the generalization capacity of a model. The underfitting and overfitting problems in many data-modeling procedures can be analyzed through the well-known bias-plus-variance decomposition of the prediction error.

The idea of randomized subspace learning used in Method I along with bootstrap seeding used in Method II seems to be useful in mitigating the above-mentioned problem; yet, not powerful enough. IVM seems to be most stable achieving Var. of 1.8248, while Methods I and II reach about 2.7 only. Such experimental results on model stability could be disappointing; however, at the same time, it could be a room for us for further improvement.

All in all, (a) of Fig. 2 depicts the performance comparisons of different models in Acc, Sp, Sn, and Mcc. As shown, Method I outperforms all other models in Acc, Sn, and Mcc, and no significant differences observed in Sp. As in (b) of Fig. 2, Methods I and II have successfully reduced FNR. Part (c) of Fig. 2 shows that Method I gives almost the same accuracy on different number of folds, while Method II accuracy noticeably varies on different number of folds. However, no significant improvement observed in Sp and FPR. Part (d) of Fig. 2 compares the model stability. IVM method clearly outperforms other models.



(a) Comparisons in Ac, Sp, Sn, and Mcc

(b) Comparisons between FPR and FNR Errors

(c) Comparisons between different methods accuracy on different folds

(d) Comparisons in StDev

Fig. 2. Model performance comparisons.

4 Conclusion

In this paper, we describe experiments with novel randomized subspace learning and bootstrap seeding techniques as an extension to our earlier work, the consensus models as well as entropy-based learning methods, to obtain better accuracy through a precise and robust learning scheme for proline CTI prediction. Our experimental results demonstrates the success of randomized subspace learning and bootstrap seeding techniques for proline CTI prediction in the well-referenced benchmark dataset. We believe this is the first time accurate computational prediction has been reported achieving Q2 accuracy above 90 percent.

Acknowledgments

The authors are grateful to Dr. Song at the ARC Centre of Excellence in Structural and Functional Microbial Genomics at Monash University, Australia, for providing us the proline cis-trans isomerization dataset, as well as her invaluable discussions.

References

[1] G. N. Ramachandran and A. K. Mitra, "An explanation for the rare occurrence of cis peptide units in proteins and polypeptides," *J. Mol. Biol.*, vol. 107, pp. 85–92, 1976.

[2] W. L. Jorgensen and J. Gao, "Cis-trans energy difference for the peptide bond in the gas phase and in aqueous solution," *J. Amer. Chem. Soc.*, vol. 110, pp. 4212–4216, 1988.

[3] G. Scherer, M. L. Kramer, M. Schutkowski, U. Reimer, and G. Fischer, "Barriers to rotation of secondary amide peptide bonds," *J. Amer. Chem. Soc.*, vol. 120, pp. 5568–5574, 1998.

[4] S. J. Eyles, "Proline not the only culprit?" *Nat. Struct. Biol.*, vol. 8, pp. 380–381, 2001.

[5] E. R. Schonbrunner and F. X. Schmid, "Peptidyl-prolyl cis-trans isomerase improves the efficiency of protein disulfide isomerase as a catalyst of protein folding," *Proc. Nat. Acad. Sci. USA*, vol. 89, pp. 4510–4513, 1992.

[6] U. Reimer, G. Scherer, M. Drewello, S. Kruber, M. Schutkowski, and G. Fischer, "Side-chain effects on peptidyl-prolyl cis/trans isomerization," *J. Molecular Biol.*, vol. 279, pp. 449–460, 2003.

[7] B. Eckert, A. Martin, J. Balbach, and F. X. Schmid, "Prolyl isomerization as a molecular timer in phage infection," *Nat. Struct. Mol. Biol.*, vol. 12, pp. 619–623, 2005.

[8] W. J. Wedemeyer, E. Welker, and H. A. Scheraga, "Proline cis-trans isomerization and protein folding," *Biochemistry*, vol. 41, pp. 14637–14644, 2002.

[9] J. Balbach and F. X. Schmid, "Proline isomerization and its catalysis in protein folding," *Mechanisms of Protein Folding*. R. H. Pain, eds., Oxford, U.K.: Oxford Univ. Press, 2000, pp. 212–249.

[10] T. E. Creighton, "Protein folding coupled to disulphidebond formation," *Mechanisms of Protein Folding*. R. H. Pain, ed., Oxford, U.K.: Oxford Univ. Press, 2000, pp. 250–278.

[11] C. Frömmel and R. Preissner, "Prediction of prolyl residues in cis-con formation in protein structures on the basis of the amino acid sequence," *FEBS Lett.*, vol. 277, pp. 159–163, 1990.

[12] M. L. Wang, W. J. Li, and W. B. Xu, "Support vector machines for predic tion of peptidyl prolyl cis/trans isomerization," *J. Peptide Res.* vol. 63, pp. 23–28, 2004.

[13] J. Song, K. Burrage, Z. Yuan, and T. Huber, "Prediction of cis/trans isomerization in proteins using PSI-BLAST profiles and secondary structure information," *BMC Bioinformatics*, vol. 7, no. 124, 2006.

[14] S. Altschul, W. Gish, W. Miller, E. Myers, and D. Lipman, "Basic local alignment search tool," *J. Molecular Biol.*, vol. 215, no. 3, pp. 403–410, 1990.

[15] T. N. Petersen, C. Lundegaard, M. Nielsen, H. Bohr, J. Bohr, S. Brunak, G. P. Gippert, and O. Lund, "Prediction of protein secondary structure at 80% accuracy," *PROTEINS: Struct., Function, Genetics*, vol. 14, pp. 17–20, 2000.

[16] D. Pahlke, D. Leitner, U. Wiedemann, and D. Labudde, "COPS-cis/trans peptide bond conformation prediction of amino acids on the basis of secondary structure information," *Bioinformatics*, vol. 21, pp. 685–686, 2005.

[17] K. Exarchos, C. Papaloukas, T. P. Exarchos, A. N. Troganis, and D. I. Fotiadis, "Prediction of cis/trans isomerization using feature selection and support vector machines," *J. Biomed. Informatics*, vol. 42, pp. 140–149, 2009.

[18] P. D. Yoo, B. B. Zhou, and A. Y. Zomaya, "Machine learning techniques for protein secondary structure prediction: An overview and evaluation," *J. Current Bioinformatics*, vol. 3, no. 2, pp. 74–86, 2008.

[19] P. D. Yoo, A. Sikder, J. Taheri, B. B. Zhou, and A. Y. Zomaya, "DomNet: Protein domain boundary prediction using enhanced general regression network and new profiles," *IEEE Trans. NanoBiosci.*, vol. 7, no. 2, pp. 172–181, Jun. 2008.

[20] P. D. Yoo, S. Ho, B. B. Zhou, and A. Y. Zomaya, "SiteSeek: Protein post-translational modification analysis using adaptive locality-effective kernel methods and new profiles," *BMC Bioinformatics*, vol. 9, no. 272, 2008.

[21] P. D. Yoo, B. B. Zhou, and A. Y. Zomaya, "A modular kernel approach for integrative analysis of protein inter-domain linker regions," *BMC Genomics BMC Genomics*, vol. 10, no. 3, p. S21, 2009.

[22] P. D. Yoo, Y. S. Ho, J. Ng, M. Charleston, N. K. Saxena, P. Yang, and A. Y. Zomaya, "Hierarchical kernel mixture models for the prediction of AIDS disease progression using HIV structural gp120 profile," *J. BMC Genomics*, vol. 11, no. 4, p. S22, 2010.

[23] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learning*, vol. 20, no. 3, p. 273, 1995.

[24] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, pp. 119–139, 1997.

[25] D. Michael, *Random Trees: An Interplay between Combinatorics and Probability*. New York, NY, USA: Springer-Verlag, 2009.

[26] P. D. Yoo, S. Muhaidat, K. Taha, J. Bentahar, and A. Shami, "Intelligent consensus modeling for proline cis-trans isomerization prediction," *IEEE/ACM Trans. Comput. Biol. Bioinform.*, vol. 11, no. 1, pp. 26–32, Jan./Feb. 2014.

[27] PISCES: a protein sequence culling server [Online]. Available: http://dunbrack.fccc.edu/PISCES.php, Nov. 2014.

[28] D. T. Jones, "Protein secondary structure prediction based on position-specific scoring matrices," *J. Molecular Biol.*, vol. 292, pp. 195–202, 1999.

[29] NCBI FTP website [Online]. Available: ftp://ftp.ncbi.nlm.nih.gov/blast/db/, Nov. 2014.

[30] J. Qian, J. Lin, N. M. Luscombe, H. Yu, and M. Gerstein, "Prediction of regu- latory networks: genome-wide identification of transcription factor targets from gene expression data," *Bioinformatics*, vol. 19, pp. 1917–1926, 2003.

[31] Weka 3: Data Mining Software in Java, Machine Learning Group, the University of Waikato, New Zealand [Online]. Available: http://www.dbs.ifi.lmu.de/~zimek/diplomathesis/implementations/EHNDs/doc/weka/classifiers/meta/ThresholdSelector.html, Nov. 2014.

Omar Y. Al-Jarrah received the BS degree in computer engineering from Yarmouk University, Jordan, in 2005, and the MS degree in computer engineering from the University of Sydney, Australia, in 2008. He is currently working toward the PhD degree at Khalifa University. From 2008 to 2012, he was at Al-ahlyiaa Amman University and IAT as an instructor. His main research interests involve machine learning, big data analytic, and knowledge discovery in various applications.

Paul D. Yoo received the PhD degree in engineering and IT from the University of Sydney (USyd) in 2008. He was a research fellow in the Centre for Distributed and High Performance Computing, at USyd from 2008 to 2009, and a PhD researcher (quantitative analysis) at the Capital Markets CRC, administered by the Australia Federal Department, for Education, Science and Training, from 2004 to 2008. He is currently a lecturer of data science and analytics in Data Science Institute, Bournemouth University, United Kingdom. He holds more than 40 prestigious journal and conference publications and is currently actively involved in editorial board, technical program committees, and review panels of the data science and analytics for international conference and journal publications such as the IEEE, ACM, and ISCB.

Kamal Taha received the PhD degree in computer science from the University of Texas at Arlington, in March 2010. He has been an assistant professor in the ECE Department, Khalifa University, Abu Dhabi, since 2010. He has over 30 refereed publications that have appeared in journals, conference proceedings, and book chapters. He was an instructor of computer science at the University of Texas at Arlington from August 2008 to August 2010. He was an engineering specialist for Seagate Technology (a leading computer disc drive manufacturer in the US) from 1996 to 2005. His current research interests include bioinformatics databases (mediators, ontologies), information retrieval in semistructured data, keyword search in XML documents, recommendation systems and social networks, and knowledge discovery and data mining. He serves as a member of the Program Committee of a number of international conferences, and he is a reviewer for a number of academic journals and conferences. He was selected by Marquis Who's Who to be included in the 2011-2012 (11th) Edition of *Who's Who in Science and Engineering*. He is a member of the IEEE.

Sami Muhaidat received the PhD degree in electrical and computer engineering from the University of Waterloo, Ontario, in 2006. From 2007 to 2008, he was an NSERC postdoctoral fellow in the Department of Electrical and Computer Engineering, University of Toronto, Canada. From 2008 to 2012, he was an assistant professor in the School of Engineering Science, Simon Fraser University, BC, Canada. He is currently an assistant professor at Khalifa University and a visiting reader in the Faculty of Engineering, University of Surrey, United Kingdom. His research focuses on advanced digital signal processing techniques for image processing and communications, machine learning, cooperative communications, vehicular communications, MIMO, and space-time coding. He has authored more than 100 journal and conference papers on these topics. He is an active senior member of the IEEE and currently serves as an editor for the *IEEE Communications Letters* and an associate editor for the *IEEE Transactions on Vehicular Technology*. He received several scholarships during the undergraduate and graduate studies. He was also a winner of the 2006 NSERC Postdoctoral Fellowship competition.

Abdallah Shami (M'03-SM'09) received the BE degree in electrical and computer engineering from the Lebanese University, Beirut, Lebanon, in 1997, and the PhD degree in electrical engineering from the Graduate School and University Center, City University of New York, New York, NY, in September 2002. In September 2002, he joined the Department of Electrical Engineering at Lakehead University, Thunder Bay, ON, Canada, as an assistant professor. Since July 2004, he has been with The University of Western Ontario, London, ON, Canada, where he is currently an associate professor in the Department of Electrical and Computer Engineering. His current research interests are in the areas of wireless/optical networking, big data, and biomedical informatics. He is a senior member of the IEEE.

Nazar Zaki received the PhD degree in computer science from the Universiti Teknologi Malaysia (UTM), Malaysia, in 2004. In 2004, he joined the College of Information Technology (CIT), United Arab Emirates University (UAEU), as an assistant professor, and in 2009 he became an associate professor. He is currently coordinating two tracks namely Intelligent Systems and Software Development. His research is mainly focuses on developing intelligent data mining algorithms to solve specific biological problems, such as protein function prediction, protein interaction network analysis, and protein functional complex detection. He has managed to raise more than 3.5 Millions of AED for research and has authored/coauthored more than 60 publications in reputable journals and conferences. He received many scholarship awards such as the CIT College Recognition Award for Excellence in Scholarship in 2007 and 2012, respectively, and the Best Paper Award in ACM Genetic and Evolutionary Computation Conference in 2011.