Carbohydrate Research 486 (2019) 107857

Contents lists available at ScienceDirect

Carbohydrate Research

journal homepage: www.elsevier.com/locate/carres

StackCBPred: A stacking based prediction of protein-carbohydrate binding sites from sequence

Suraj Gattani[a,1], Avdesh Mishra[b,1], Md Tamjidul Hoque[a,*]

[a] Department of Computer Science, University of New Orleans, 2000 Lakeshore Dr, New Orleans, LA, 70148, USA
[b] Department of Electrical Engineering and Computer Science, Texas A&M University-Kingsville, 700 University Blvd, Kingsville, TX, 78363, USA

ARTICLE INFO

Keywords:
Protein-carbohydrate binding
Binding prediction
Machine learning
Stacking

ABSTRACT

Carbohydrate-binding proteins play vital roles in many important biological processes. The study of these protein-carbohydrate interactions, at residue level, is useful in treating many critical diseases. Analyzing the local sequential environments of the binding and non-binding regions to predict the protein-carbohydrate binding sites is one of the challenging problems in molecular and computational biology. Existing experimental methods for identifying protein-carbohydrate binding sites are laborious and expensive. Thus, prediction of such binding sites, directly from sequences, using computational methods, can be useful to fast annotate the binding sites and guide the experimental process. Because the number of carbohydrate-binding residues is significantly lower than the number of non-carbohydrate-binding residues, most of the methods developed for the prediction of protein-carbohydrate binding sites are biased towards over predicting the negative class (or non-carbohydrate-binding). Here, we propose a balanced predictor, called StackCBPred, which utilizes features, extracted from evolution-driven sequence profile, called the position-specific scoring matrix (PSSM) and several predicted structural properties of amino acids to effectively train a Stacking-based machine learning method for the accurate prediction of protein-carbohydrate binding sites (https://bmll.cs.uno.edu/).

1. Introduction

Organisms need four types of molecules: nucleic acids, proteins, carbohydrates (or polysaccharides) and lipids for life, which are usually referred to as the molecules of life. Carbohydrates are often considered as the third important molecule of life, after DNA and proteins [1]. Carbohydrate interacts with many different protein families which include lectins, antibodies, sugar transporters and enzymes [2]. Protein-carbohydrate interactions are responsible for various biological processes, including intercellular signaling, cellular adhesion, cellular recognition, protein folding, subcellular localization, ligand recognition and developmental process [3–5]. In fact, carbohydrate of one or the other type generally covers living cells in all organisms [6]. These carbohydrates play important roles in the defense of human cells against pathogens [7]. Moreover, some pathogens such as influenza use these carbohydrates on the outside of the human cell to gain entry [8]. The proteins, which recognize and bind to the cell-surface carbohydrates, are useful as biomarkers or drug targets [8–11]. The study of protein-carbohydrate interactions is usually carried out by experimental techniques including X-ray crystallography, nuclear magnetic resonance (NMR) spectroscopy study, molecular modeling, fluorescence spectrometry, and dual polarization interferometry. However, protein-carbohydrate interactions are challenging to study experimentally because of the weak binding affinity and synthetic complexity of individual carbohydrates [6]. Therefore, the prediction of protein-carbohydrate interactions through a computational approach becomes essential. This motivates us to develop an effective computational predictor for effective identification and characterization of protein-carbohydrate binding sites.

The study of protein-carbohydrate interactions using computational methods mainly focuses on locating the sites of proteins that bind to carbohydrates. One of the promising computational techniques is docking. Docking methods are often used to predict the orientation of the carbohydrate in the binding site [2]. On the other hand [12], proposed a first bioinformatics approach for predicting protein-carbohydrate binding sites from a known protein structure. In their work, six parameters of amino acids were evaluated, which include solvation potential, residue propensity, hydrophobicity, planarity, protrusion, and relative accessible surface area. A simple combination of three of the parameters (residue propensity, protrusion, and relative accessible

* Corresponding author.
E-mail addresses: sggattan@uno.edu (S. Gattani), avdesh.mishra@tamuk.edu (A. Mishra), thoque@uno.edu (M.T. Hoque).
[1] These authors contributed equally to this work as the first authors.

https://doi.org/10.1016/j.carres.2019.107857
Received 25 June 2019; Received in revised form 5 October 2019; Accepted 23 October 2019
Available online 24 October 2019
0008-6215/ Published by Elsevier Ltd.

surface area) out of six was found to distinguish the observed binding sites with an overall accuracy of 65% for a set of 40 protein-carbohydrate complexes. A continuous surface pocket interacting with protein-probes was considered binding sites. Nassif.et al. proposed a glucose-binding site classifier that considers the sugar-binding pocket as a spherical spatio-chemical environment and represents it as a vector of geometric and chemical features which includes charges, hydrophobicity, hydrogen bonding and more [13]. They employed random decision forests for feature selection and used selected geometric and chemical features to train the support vector machines (SVM) for predicting protein-glucose binding sites. Tsai.et al. predicted binding sites by employing three-dimensional probability density distribution of interacting atoms in protein surfaces as input to the neural networks and SVM [14]. In the recent past, an energy-based approach for the identification and analysis of binding sites residues in protein-carbohydrate complexes has been proposed [15]. Through this study, it was found that 3.3% of residues are identified as binding sites in protein-carbohydrate complexes whereas the binding site residues in protein-protein, protein-RNA, and protein-DNA complexes are 10.8%, 7.6%, and 8.7% respectively. Furthermore, the binding propensity analysis performed in this study indicates the dominance of Tryptophan (TRP) amino acid to interact with the carbohydrates through aromatic-aromatic interactions. More recently, Shanmugam et al. proposed a method to identify and analyze the residues, which are involved in both the folding and binding of protein-carbohydrate complexes [16]. Stabilizing residues were identified by using the knowledge of hydrophobicity, long-range interactions, and conservations, as well as binding site residues, were identified using a distance cutoff of 3.5 Å between heavy atoms in protein and ligand. Residues, which were common in stabilizing and binding, were termed as key residues. Some of the interesting findings of the work indicate that most of the key residues are present in β-strands and polar and charged residues have a high tendency to serve as key residues.

The structure-based methods discussed above, rely on protein structures that are often not available, which makes the sequence-based method inevitable. The first sequence-based method for protein-carbohydrate binding sites prediction was developed by Malik and Ahmad in 2007 [8]. In their work, Malik and Ahmad used only the evolutionary attributes called PSSM as input to the neural network to create a predictive model. Their method achieved an average of 87% sensitivity and 23% specificity while tested by leave-one-out technique on a dataset of 40 protein-carbohydrate complexes. After a year less than a decade, Taherzadeh et al. [6] proposed a method, called SPRINT-CBH, which used PSSM profiles with additional information on sequence and predicted solvent accessible surface area as features to develop an SVM based predictor in 2016.

As reported, SPRINT-CBH achieved an average of 18.8% sensitivity and 99.6% specificity while tested using 10-fold cross-validation (CV) on a dataset of 102 protein-carbohydrate complexes and 22.3% sensitivity and 98.8% specificity while tested using independent test set of 50 protein-carbohydrate complexes. Both the aforementioned methods suffer from the problem of imbalanced prediction accuracies. These methods either yield high sensitivity and low specificity or vice versa. Thus, the existing methods are limited in their ability to effectively predict binding sites and explain how protein-carbohydrate interaction occurs. Therefore, it becomes essential to identify new features and effective machine learning techniques that can help in improved binding site prediction as well as help interpret protein-carbohydrate interactions. One of the reasons why both predictors suffer from imbalance prediction accuracies could be that some of the features used in these studies might be negatively impacting the sensitivity of the predictor. In our study, we found that the Secondary Structure features were affecting the prediction accuracy negatively. Therefore, we excluded Secondary Structure features from our feature set. We strongly believe lack of exploration of different features and using the most relevant features to obtain the predictions makes a significant improvement in the quality of the predictor.

While there are still very few methods for predicting protein-carbohydrate binding sites, many other methods have been established for the binding site and binding proteins prediction in the area of protein-protein [17–20], protein-peptide [21–24], protein-DNA [25–28], protein-RNA [29–32] and protein-ligand [33–36] interactions. Several of the aforementioned sequence-based methods have shown that the use of evolution-derived and predicted sequence and structure-based features can significantly improve the overall performance of the binding sites prediction.

In this study, we investigated different descriptors, which include information extracted from the evolutionary profile as well as predicted sequence and structural properties. Moreover, we examined various machine learning approaches to develop a sequence-based unbiased and balanced predictor of non-covalent protein-carbohydrate binding sites. Useful feature groups were selected to build a Stacking-based classifier called StackCBPred. The StackCBPred was trained and cross-validated by 100 carbohydrate-binding proteins and independently tested by two different test sets containing 50 and 88 proteins with known high-resolution protein-carbohydrate complex structures, respectively. As the dataset contain significantly more non-binding residues than binding residues, StackCBPred was trained with a balanced dataset obtained by employing the under-sampling technique to design a more balanced predictor. The development of StackCBPred offered a significant improvement in sensitivity and balanced accuracy based on the benchmark and independent test data when compared to the existing sequence-based binding predictor. We believe that the superior performance of StackCBPred will motivate the researchers to use this method to identify protein-carbohydrate binding sites directly from sequence and utilize the outcomes for drug targeting. In addition, the stacking-based machine learning technique and features proposed in this work could be applied to solve various other biologically important problems.

2. Methods

In this section, we describe the approach taken to prepare benchmark and independent test data sets, feature extraction, feature selection, performance evaluation, and machine learning framework development.

2.1. Dataset

We collected the benchmark dataset [6] that contains a total of 102 high-resolution carbohydrate-binding protein sequences. However, in our implementation, we only used 100 high-resolution carbohydrate-binding protein sequences for training and cross-validation as two of the sequences contain non-standard amino acid and the physicochemical properties of the non-standard amino acids could not be obtained. Furthermore, we collected an independent test dataset [6] to compare the performance of StackCBPred with the existing predictor. This dataset consists of 50 high-resolution carbohydrate-binding protein sequences, of which 49 were used in our implementation, discarding one for having the nonstandard amino acids in the sequence information. Here and after we represent this test dataset as TS49.

From the benchmark dataset of 100 sequences, we obtained a total of 26,986 residues, of which, 1028 residues are binding, and the rest are non-binding. To avoid bias caused by many non-binding residues, a balanced dataset was prepared following an under-sampling approach [37] by randomly selecting a number of non-binding residues equal to the number of binding residues. This resulted in a benchmark dataset, which consists of 1028 binding and an equal number of non-binding residues. From TS49 sequences, we obtained a total of 13,738 residues of which 508 residues are binding and the rest are non-binding. Using a similar under-sampling approach as above, a balanced independent test set was prepared which consists of 508 binding and an equal number of

non-binding residues.

To further test the performance of our predictor, we collected an additional dataset PROCARB604 from PROCARB [38] database. The proteins whose ID's matched to the protein ID's that were present in either the benchmark or the independent test dataset mentioned above were removed from this new dataset. Next, the redundant proteins with a sequence identity cutoff of ≥ 30% according to BLAST-CLUST [39] were removed. Finally, the dataset, which consists of 88 protein-carbohydrate complexes was obtained. Here and after we represent this dataset as TS88. This new TS88 dataset consists of 688 binding residues. Using an under-sampling approach, we prepared a balanced dataset which contains 688 binding and an equal number of non-binding residues. In addition to evaluating our predictor on balanced validation and independent test datasets, we evaluated our predictor on imbalanced (full) validation and independent test datasets.

2.2. Feature extraction

We collected various useful features which include information extracted from the evolutionary profile as well as predicted sequence and structural properties of proteins, which we described in this section.

2.2.1. Position specific scoring matrix (PSSM) and monogram (MG)

PSSM captures the evolution derived information in proteins. Evolutionary information is very impactful for protein function annotation in biological analysis and is widely used in many studies [21,26,40–43]. Furthermore, evolutionarily conserved residues are found to play crucial functional roles such as binding [44]. For this study, we obtained the normalized PSSM values for every residue in protein sequence from DisPredict2 [41,42] program. DisPredict2 internally executes three iterations of PSI-BLAST [21,45] against NCBI's non-redundant database to generate a PSSM profile and subsequently converts it to normalized PSSM by dividing each value by a value of 9. PSSM is a matrix of L × 20 dimensions, where L is the length of the protein. The rows in PSSM represent the position of amino acid in the sequence, and the columns represent the 20 standard amino acid types. Hence, every residue in the protein sequence is encoded by a 20-dimensional feature vector. In addition, the PSSM score was further extended to compute monogram feature [46,47], which is obtained by taking the sum of the scores over the length of the protein for 20 standard amino acid types. This resulted in 1 feature for every amino acid.

2.2.2. Accessible surface area (ASA) and secondary structure (SS)

ASA and SS are predicted structural features that are found to be highly effective for binding sites prediction. We used the DisPredict2 program to obtain predicted ASA and SS probabilities for helix, coil, and beta-sheet at the residue level. DisPredict2 internally uses a program called SPINE-X [48] to predict ASA and SS probabilities directly from the protein sequence.

2.2.3. Half sphere exposure (HSE) and torsion angles

HSE is a measure of protein solvent exposure that was first introduced in Ref. [49]. HSE measures how buried amino acid residues are in protein conformation. The calculation of HSE is obtained by dividing a contact number (CN) sphere into two halves by the plane perpendicular to the Cβ-Cα vector. This simple division of the CN sphere produces two different measures, called HSE-up and HSE-down. In this study, we used these two measures as features that were extracted from the SPIDER3 program [50–52]. Additionally, protein backbone structure can be described by torsion angles Phi (φ) and Psi (ψ). This local structure descriptor is important for understanding and predicting protein structure, function, and interactions. In our study, we employed predicted φ and ψ angles as features that were also extracted from SPIDER3 program.

2.2.4. Physiochemical properties

We obtained seven representative physiochemical attributes of the amino acids, which include steric parameters, hydrophobicity, volume, polarizability, isoelectric point, helix probability, and sheet probability [53]. As these features are inherently encoded within DisPredict2, we directly extracted these features from the DisPredict2 [41].

2.2.5. Molecular recognition features (MoRFs)

Post-translational modifications (PTMs) can induce disorder-to-order transitions of intrinsically disordered proteins (IDPs). IDPs can transition from disorder to order due to binding to other proteins, nucleic acids, lipids, carbohydrates and other small molecules [54,55]. MoRFs are critical to the biological function of IDPs located within long disordered protein sequences [56–58]. Thus, to inherently capture functional properties of IDPs which may bind to carbohydrates, we employed a single predicted MoRFs score as a feature in this work. We obtain the MoRFs feature from OPAL [57].

2.3. Performance evaluation

The performance of the StackCBPred was evaluated by 10-fold CV as well as using the independent test. In 10-fold CV, the dataset is segmented into 10 parts, which are each of about equal size. When a fold is set aside for testing, the other 9 folds are used to train the classifier. This process is repeated until each fold has been set aside once for testing and then the test accuracies of each fold are combined to find the average [59]. On the other hand, to perform the independent test, the classifier is trained with the validation dataset and then tested using the independent test dataset. We used various performance evaluation metrics listed in Table 1 to test the accuracy of our proposed method as well as to compare it with the existing method.

In addition, we used AUC and ROC performance evaluation metrics. AUC is the area under the receiver operating characteristics (ROC) curve and is used to evaluate a predictor to see how well it separates two classes of information, which is, in this case, carbohydrate-binding and non-binding residues.

2.4. Framework of StackCBPred

The idea of stacking based machine learning technique [60] which

Table 1
Name and definition of the evaluation metric.

Name of Metric	Definition
True Positive (TP)	Correctly predicted carbohydrate-binding residues
True Negative (TN)	Correctly predicted non-carbohydrate binding residues
False Positive (FP)	Incorrectly predicted carbohydrate-binding residues
False Negative (FN)	Incorrectly predicted non-carbohydrate binding residues
Recall/Sensitivity/True Positive Rate (TPR)	$\frac{TP}{TP+FN}$
Specificity/True Negative Rate (TNR)	$\frac{TN}{TN+FP}$
Fall-out Rate (FOR)/False Positive Rate (FPR)	$\frac{FP}{FP+TN}$
Miss Rate (MR)/False Negative Rate (FNR)	$\frac{FN}{FN+TP}$
Accuracy (ACC)	$\frac{TP+TN}{FP+TP+TN+FN}$
Balanced Accuracy (BACC)	$\frac{1}{2}\left(\frac{TP}{TP+FN}+\frac{TN}{TN+FP}\right)$
Precision (Prec.)	$\frac{TP}{TP+FP}$
F1 score (Harmonic mean of precision and recall)	$\frac{2TP}{2TP+FP+FN}$
Mathews Correlation Coefficient (MCC)	$\frac{(TP\times TN)-(FP\times FN)}{\sqrt{(TP+FN)\times(TP+FP)\times(TN+FP)\times(TN+FN)}}$

has recently been successfully applied to solve some interesting bioinformatics problems [21,26,61–63] is utilized in this work to develop the StackCBPred predictor for carbohydrate-binding sites prediction. Stacking is an ensemble approach, which obtains the information from multiple models and aggregates them to form a new model. In stacking, the information gained from more than one predictive model minimizes the generalization error rate and yields more accurate results.

The stacking framework includes two-stages of learners. The classifiers of the first-stage are called base-classifiers. More than one base-classifiers are employed in the first stage. Likewise, the classifiers of the second-stage are called meta-classifiers. Using meta-classifier, the prediction probabilities from the base-classifiers are combined to reduce the generalization error. To supply the meta-classifier with significant information on the problem space, the classifiers that are different from one another based on their underlying operating principle are used as the base-classifiers.

To find the base-classifiers and meta-classifiers to use in the first and second-stage of stacking framework, we examined eight different machine learning algorithms: (a) Support Vector Machines (SVM) [64], (b) Gradient Boosting Classifier (GBC) [65], (c) Bagging Classifier (BAG) [66], (d) Extra Tree Classifier (ETC) [67], (e) Random Decision Forest (RDF) [68], (f) K-Nearest Neighbor (KNN) [69], (g) Logistic Regression (LOGREG) [59,70] and (h) XGBoost (XGB) [71].

Algorithms mentioned above are built and optimized using Scikit-learn [72]. To select the algorithms to be used as the base-classifiers for the stacked model, we evaluate four different combinations of base-classifier which are:

1. Model-1: includes SVM, LOGREG, KNN, and ETC.
2. Model-2: includes SVM, LOGREG, KNN, and RDF.
3. Model-3: includes SVM, LOGREG, KNN, and BAG.
4. Model-4: includes GBC, LOGREG, and KNN.

Model-1, Model-2, and Model-3 are constructed to include classifiers that are different from each other based on the underlying principles of learning. Here, the tree-based classifiers ETC, RDF and BAG are individually combined with the other three classifiers, SVM, LOGREG and KNN to learn different information from the problem-space. On the other hand, Model-4 is formed by the pair-wise correlation analysis of the residue-wise probabilities given by the individual classifiers. Three of the classifiers, with the least Pearson correlation coefficient, are selected as base-classifiers. For all the above combinations, SVM is used as a meta-classifier. The 10-fold CVs of the above four combinations indicate that the Model-1, when combined with SVM gives the best performance. Therefore, we employ four classifiers SVM, LOGREG, KNN and ETC as base classifiers and SVM as meta-classifier in the StackCBPred framework. In StackCBPred, the binding and non-binding probabilities generated by the four base-classifiers are combined with original 33 features which include PSSM, MG, ASA, Physiochemical properties, Phi & Psi angles, and HSE up & HSE down and are given as input features to the meta-classifier which eventually predict binding and non-binding residues. Fig. 1 illustrates the prediction framework of StackCBPred.

In order to find the base-classifiers to use in the first stage and the meta-classifier to use in the second-stage of stacking framework, we explored eight different machine learning algorithms and they are:

i) *SVM:* We employed SVM [64] with the radial basis function (RBF) kernel as one of the classifiers to be used in the stacking framework. SVM classifies by maximizing the separating hyperplane between two classes and penalizes the instances on the wrong side of the decision boundary. The performance of SVM with the RBF kernel relies on two parameters *C*, and γ. The RBF kernel parameter γ and the cost parameter *C* are optimized to achieve the best 10-fold CV balanced accuracy using a grid search [73] technique. The optimal values of the parameters of the SVM were found to be $C = 2^{1.24}$ and $\gamma = 2^{-8.75}$.

ii) *LOGREG:* We implemented LOGREG [59,70] with *L2* regularization as another classifier to be used in the staking framework. It measures the relationship between the dependent categorical variable (in our case: a carbohydrate-binding or non-carbohydrate-binding) and one or more independent variables by generating an estimation probability using logistic regression. The parameter, *C* which controls the regularization strength is optimized to achieve the best 10-fold CV balanced accuracy using grid search [73]. In our implementation, the optimal value of the parameter, C was found to be 2.3784.

iii) *ETC:* We employed an extremely randomized tree or ETC [67] as another classifier to be used in stacking framework. ETC fits

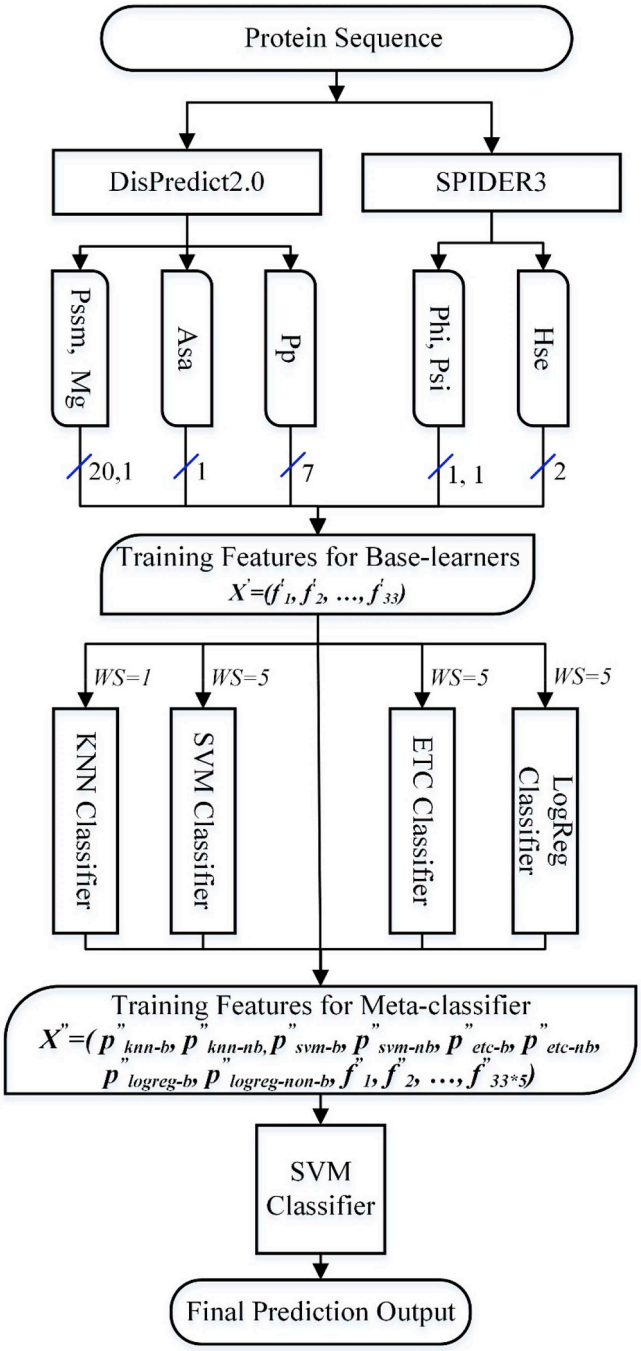


Fig. 1. Illustration of the framework of the final predictor, StackCBPred, which is principally the Model-1 with another version of SVM in the meta layer.

several randomized decision trees from the data sample and uses averaging to improve the prediction accuracy and control over-fitting. We constructed the ETC model with 1000 trees and the quality of a split was assessed by the Gini impurity index.

iv) *RDF:* RDF [68] constructs a multitude of decision trees on various sub-samples of the dataset and outputs the mean prediction of the decision trees to improve the predictive accuracy and control over-fitting. In our implementation of the RDF, we used bootstrap samples to construct 1000 trees in the forest.

v) *KNN:* KNN [69] operates by learning from the K number of training samples closest in distance to the target point in the feature space. The classification decision is computed from the majority votes coming from the neighbors. In this work, the value of K was set to 9 and all the neighbors were weighted uniformly.

vi) *LibD3C:* LibD3C [74] is based on k-means clustering and a dynamic selection strategy. It uses various subset classifiers to select a final set of ensemble classifiers to compute the decision and make the appropriate prediction.

vii) *BAG:* BAG [66]method forms a class of algorithms that builds several instances of a base classifier/estimator on random subsets of the training samples and subsequently aggregates their individual predictions to yield a final prediction. It is useful for reducing variance in the prediction. In our study, BAG classifier was fit on multiple subsets of data with the repetitions using 1000 decision trees, and the outputs were combined by weighted averaging.

viii) *GBC*: GBC [65] involves three elements: (a) a loss function to be optimized, (b) a weak learner to make predictions and (c) an additive model to add weak learners to minimize the loss function. The objective of GBC is to minimize the loss of the model by adding weak learners in a stage-wise fashion using a procedure similar to gradient descent. The existing weak learners in the model are remained unchanged while adding new weak learners. The output from the new learner is added to the output of the existing sequence of learners in an effort to correct or improve the final output of the model. Here, we used 1000 bosting stages where a regression tree was fit on the negative gradient of the deviance loss function. The learning rate and the maximum depth of each regression tree were set to 0.1 and 3, respectively.

ix) XGB: As GBC, XGB [71] also follows the principle of gradient boosting. However, XGB uses a more regularized model formalization to control over-fitting, which results in better performance. In addition to better performance, XGB is designed to provide higher computational speed. In our implementation of the XGB, we used 100 bosting stages with a soft prob learning objective, where the number of classes was set to 2 as we are dealing with a binary classification problem of carbohydrate-binding and non-carbohydrate-binding residues. The values of the additional parameters: learning rate, maximum depth, minimum child weight, and subsample ratio were set to 0.1, 3, 5 and 0.9, respectively.

3. Results

In this section, we first demonstrate the results of the feature selection and optimal window selection. Then, we present the performance comparison of potential base-classifiers and stacked models. Finally, we report the performance of StackCBPred on the benchmark dataset and the independent test datasets and subsequently compare it with the existing method. We would like to emphasize that balanced datasets are utilized for training, cross-validation, and independent test unless otherwise indicated.

3.1. Feature selection

To identify the features that support the performance of the classifier, we applied various feature selection methods such as incremental feature selection (IFS), mRMR, and MRMD. mRMR (minimum Redundancy Maximum Relevance) [75] uses the F-statistic and correlation to calculate redundancy & relevance and select the best combination of features. MRMD [76] decides on the basis of Pearson's correlation coefficient to measure relevance and Euclidean distance (ED), Cosine distance (CD) and Tanimoto (TO) to calculate redundancy. IFS begins with the empty feature set and a feature group is added to the feature set if the addition of the feature group improves the performance of the predictor. In case, the accuracy of the predictor is reduced by adding the new feature group, this feature group is discarded, and a new feature group is tested in an iterative fashion. To obtain the best features through IFS we used the GBC predictor on benchmark dataset to train and TS49 dataset to test. Table 2 shows the comparison between the results obtained by SVM through 10-fold CV on the benchmark dataset while using different feature selection methods. From Table 2., it is evident that IFS provides the best results for all the performance metrics, except for sensitivity. The MRMD approach yields the highest sensitivity, however, shows poor results for other performance metrics. Therefore, we selected IFS as a feature selection method in this work. We initially collected thirty-nine features (see Section 2.2), of which, we discarded six features based on IFS. The three secondary structure features and three MoRFs features were removed as these features did not help improve the performance of the predictor.

Table 2
Performance comparison of SVM models obtained using features selected through various feature selection techniques with optimal sliding window size on benchmark dataset through a 10-fold CV.

Metric/Method	mRMR	MRMD	IFS
Sensitivity	0.734	**0.760**	0.737
Specificity	0.750	0.723	**0.762**
Bal. Accuracy	0.742	0.742	**0.750**
Accuracy	0.742	0.742	**0.750**
Precision	0.746	0.733	**0.756**
F1 score	0.740	**0.746**	**0.746**
MCC	0.485	0.484	**0.499**

Best scores are boldfaced

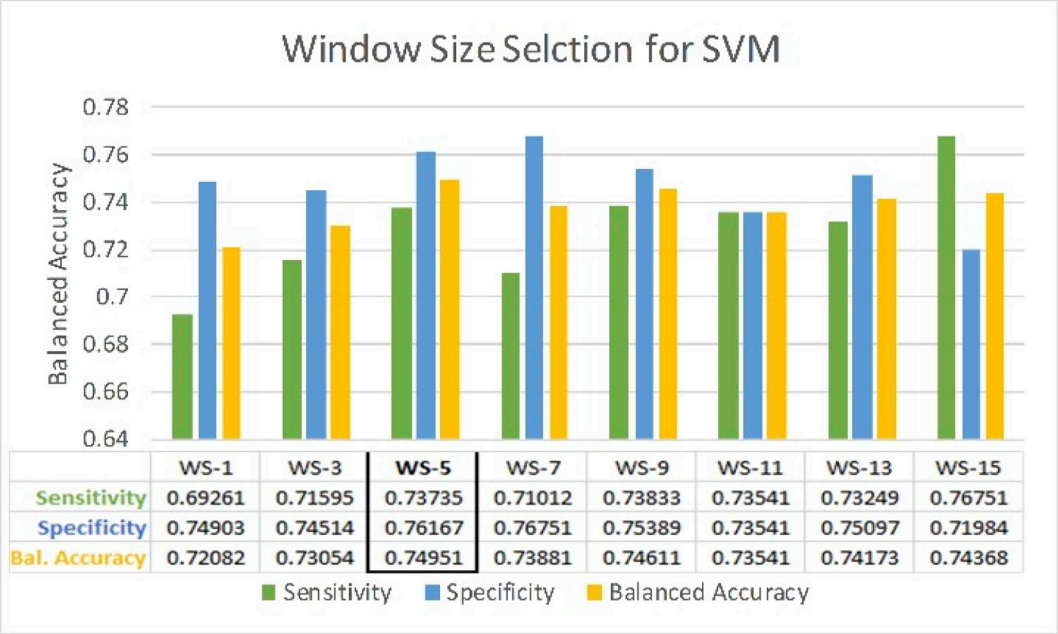
3.2. Window selection

The optimal size of the sliding window (W) was searched to determine the number of residues around a target residue, which can moderate the interaction between protein and carbohydrate. We designed 8 different models of every machine learning classifiers with 8 different window sizes (1, 3, 5, 7, 9, 11, 13 and 15). Window size for which the classifier yields the highest 10-fold CV balanced accuracy on benchmark dataset was selected as the optimal window size for that classifier. We found that the optimal window size for different classifiers is different. For example, the optimal window size for the SVM was found to be 5 (see Fig. 2) whereas, for the KNN it was 1. In this study, the optimal window size for every classifier was separately identified to design an accurate and effective predictor.

3.3. Selection of the base and meta-classifiers

To select the methods for base and meta-classifiers, we examined the performance of eight different machine learning methods, BAG, ETC, LOGREG, KNN, RDF, GBC, XGB and SVM on the benchmark dataset using 10-fold CV. The performance comparison of the classifiers is shown in Table 3.

Table 3 shows that the optimized SVM with RBF-kernel provides the highest performance in terms of specificity, fall out rate, balanced accuracy, accuracy, precision, and MCC, among all the classifiers examined in this application. Moreover, the sensitivity and miss rate is highest for the XGB and F1 score is highest for RDF. Similarly, it is



	WS-1	WS-3	WS-5	WS-7	WS-9	WS-11	WS-13	WS-15
Sensitivity	0.69261	0.71595	0.73735	0.71012	0.73833	0.73541	0.73249	0.76751
Specificity	0.74903	0.74514	0.76167	0.76751	0.75389	0.73541	0.75097	0.71984
Bal. Accuracy	0.72082	0.73054	0.74951	0.73881	0.74611	0.73541	0.74173	0.74368

Fig. 2. Performance comparison of SVM based models created from different sliding window sizes. The sensitivity, specificity and balanced accuracy are reported. The optimal size of the window and the corresponding performance scores are marked by a black rectangle. The optimal window size was selected based on the highest 10-fold CV balanced accuracy.

evident that the performance of tree-based ensemble methods, BAG, ETC, RDF, GBC, and XGB are close to SVM.

The balanced accuracy of these tree-based methods differs from each other and SVM only by about 1–4%. However, the balanced accuracies of LOGREG and KNN are 7.31% and 22.79% lower than the SVM, respectively. Moreover, the learning principles of LOGREG, KNN, and SVM are different from each other.

Following the guidelines of base-classifier selection based on different underlying principles, we initially selected SVM, LOGREG, and KNN as three of the base classifiers. Then, we added one tree-based ensemble method out of five methods, BAG, ETC, RDF, GBC, and XGB, at a time as the fourth base-classifier and formulated five different combinations. For all the combinations, the meta-classifier is SVM. Out of five combinations, we present the performance of the top three combinations namely Model-1, Model-2, and Model-3 in Table 5.

Moreover, we created an additional stacked model following the guidelines of base-classifier selection based on the low Pearson correlation coefficient. We computed the Pearson correlation coefficient (ρ) between the two sets of probabilities given by two classifiers using equation (1).

$$\rho = \frac{\sum XY}{\sqrt{\sum X^2 \sum Y^2}} \quad (1)$$

The principle of stacking states that it is preferable to use learners that are thinly correlated in the first-stage to obtain better performance at the second-stage [21]. To select thinly correlated methods, we performed a pair-wise correlation analysis of the residue-wise probabilities between the classifiers. To obtain residue-wise probabilities, the classifiers were trained on a benchmark dataset and the probabilities for each residue in the TS49 test set were obtained through an independent test. The results of these correlations are shown in Table 4.

Since the SVM was found to be the top-performing method from the above comparison between different classifiers, it was selected as a meta-classifier to create the fourth combination (i.e., Model-4). Next,

Table 4
Pair-wise correlation analysis of the probability distribution given by the base-classifiers on TS49.

Classifiers	ETC	GBC	KNN	LOG REG	RDF	BAG	SVM	XGB
ETC	–	0.918	0.881	0.946	0.997	0.989	0.875	0.868
GBC	–	–	**0.762**	0.922	0.924	0.936	**0.782**	0.727
KNN	–	–	–	**0.816**	0.879	0.861	0.831	0.804
LOG REG	–	–	–	–	0.951	0.953	0.824	0.774
RDF	–	–	–	–	–	0.994	0.875	0.878
BAG	–	–	–	–	–	–	0.863	0.855
SVM	–	–	–	–	–	–	–	0.809

Identified least pair-wise correlation scores are boldfaced.

Table 3
Comparisons of various machine learning algorithms on the benchmark dataset using 10-fold CV.

Metric/Method	BAG	ETC	LOGREG	LibD3C	KNN	RDF	GBC	XGB	SVM
Sens.	0.743	0.743	0.718	0.668	0.636	0.751	0.733	**0.763**	0.737
Spec.	0.728	0.744	0.679	0.625	0.585	0.745	0.715	0.706	**0.762**
FOR	0.272	0.256	0.321	0.374	0.415	0.255	0.285	0.294	**0.238**
MR	0.247	0.257	0.282	0.331	0.364	0.249	0.268	**0.237**	0.263
BACC	0.740	0.744	0.698	0.646	0.610	0.748	0.724	0.734	**0.750**
ACC	0.740	0.744	0.698	0.646	0.610	0.748	0.724	0.734	**0.750**
Prec.	0.734	0.744	0.691	0.640	0.605	0.747	0.720	0.722	**0.756**
F1	0.744	0.744	0.704	0.654	0.620	**0.749**	0.726	0.742	0.746
MCC	0.481	0.487	0.397	0.294	0.221	0.496	0.448	0.470	**0.499**

Best score values are boldfaced.

Table 5
Comparisons of stacked models with a different set of base classifiers on benchmark dataset through a 10-fold CV.

Metric/Method	Model-1	Model-2	Model-3	Model-4
Sensitivity	**0.861**	0.855	0.857	0.859
Specificity	**0.859**	0.859	0.859	0.859
Fall Out Rate	**0.141**	0.141	0.141	0.141
Miss Rate	**0.139**	0.145	0.143	0.141
Bal. Accuracy	**0.860**	0.857	0.858	0.859
Accuracy	**0.860**	0.857	0.858	0.859
Precision	**0.859**	0.858	0.859	0.859
F1 score	**0.860**	0.857	0.858	0.859
MCC	**0.720**	0.714	0.716	0.718

Best score values are boldfaced.

the method which is least correlated with SVM was identified. From Table 4, we can see that the SVM is least correlated with the GBC with a correlation coefficient of 0.782. Thus, GBC was selected as the first base-classifier. Consequently, the next method which is least correlated with GBC was identified. Again, from Table 4, we found that GBC is least correlated with XGB. However, as both XGB and GBC are based on boosting principle, instead of selecting XGB, next least correlated method was identified. The next least correlated method to GBC was found to be KNN with a correlation coefficient of 0.762. Thus, the KNN was selected as the second base-classifier. Successively, the least correlated method to KNN was identified. Table 4 shows that the least correlated method to KNN excluding GBC and XGB is LOGREG with a correlation coefficient of 0.816. The GBC and XGB were excluded because GBC was already selected as one of the base-classifier and XGB follows the same principle of boosting as GBC.

Finally, with the above approach GBC, KNN and LOGREG were selected as base-classifiers to create Model-4. The performance of Model-4 and its comparison with other models is shown in Table 5.

Table 5 shows that the Model-1, which includes SVM, LOGREG, KNN, and ETC as base-classifier and another version of SVM as meta-classifier, provides the highest performance. Thus, we select Model-1 as our final stacking model.

3.4. Performance comparison on benchmark dataset

Here, we compute the performance of StackCBPred using a 10-fold CV on the benchmark dataset. While performing 10-fold CV the training of the StackCBPred was done using a balanced number of samples whereas, the testing was performed using a balanced as well as an imbalanced number of samples, respectively. Testing using the imbalanced number of samples in 10-fold CV was performed so that the results could be directly compared to SPRINT-CBH. Table 6 shows the performance comparison of StackCBPred and SPRINTCBH. The quantities for all the evaluation metrics for SPRINT-CBH are obtained from Taherzadeh et al. [6].

From Table 6, we observed that the performance of SPRINT-CBH is biased more towards the negative class (non-carbohydrate binding) because of which the specificity (98.9%) is extremely high and the sensitivity (18%) is extremely low. When the test data is highly imbalanced, it is easy to achieve high overall accuracy (ACC) simply by predicting every test data point as the majority class, which is what we can see from the result of SPRINT-CBH in Table 6. Balanced accuracy, which avoids inflated performance estimates on imbalanced datasets would be a proper measure of accuracy.

However, the balanced accuracy of SPRINT-CBH was not reported in the literature. We computed the balanced accuracy of SPRINT-CBH by utilizing the expression of balanced accuracy provided in Table 1. Moreover, the main goal of the carbohydrate-binding site prediction is to predict the binding sites accurately. However, due to the low sensitivity of 18%, the SPRINT-CBH bears the risk of not identifying the binding sites accurately. On the other hand, StackCBPred can predict the binding sites more accurately than the SPRINT-CBH based on the sensitivity and balanced accuracy scores as shown in Table 6. The sensitivity of the StackCBPred is 66.5% and 86.1% for the imbalanced and balanced number of samples used in testing through a 10-fold CV. Additionally, the balanced accuracy of the StackCBPred is 66.5% and 86% for the imbalanced and balanced number of samples used in testing through a 10-fold CV.

The StackCBPred attains 13.68% improvement in balanced accuracy over SPRINT-CBH while, tested using an imbalanced test set. These results indicate that StackCBPred can predict the binding sites more accurately compared to the SPRINT-CBH.

Table 6
Comparisons of StackCBPred with SPRINT-CBH on the benchmark dataset.

Methods		Sens.	Spec.	BACC	ACC	MCC
SPRINT-CBH	Imbalanced	0.180	0.989	0.585	0.950	0.250
StackCBPred	Imbalanced	0.665	0.664	0.665	0.664	0.134
	Balanced	0.861	0.859	0.860	0.860	0.720

3.5. Performance comparison using independent test datasets

In this section, we further examine the performance of StackCBPred by performing an independent test on two independent test datasets, TS49 and TS88. The TS49 dataset was recently constructed by Taherzadeh et al. [6] to test the performance of the carbohydrate-binding site predictor, called SPRINT-CBH. However, the TS88 dataset was collected in this study to further test the robustness of StackCBPred. To test using TS49 and TS88, StackCBPred was first trained on a balanced benchmark dataset and simultaneously tested on both the independent test datasets. Table 7 lists the predictive results of StackCBPred and SPRINT-CBH on the TS49 test set.

Table 7 indicates that the StackCBPred outperforms SPRINT-CBH by 42.16% and 80.72% based on sensitivity while, tested on the imbalanced and balanced TS49 test set, respectively. Similarly, StackCBPred attains 2.59% and 22.53% improvement in balanced accuracy over SPRINT-CBH while, tested using an imbalanced and balanced TS49 test set, respectively. It is to be noted that the main goal here is to predict carbohydrate-binding sites thus, higher sensitivity is preferable.

The results in Table 7 also indicate that the sensitivity of StackCBPred improves from 55.3% to 70.3% and the balanced accuracy improves from 67.4% to 80.5% while, the number of carbohydrate-binding and non-binding residues are balanced in the TS49 test set. The improved sensitivity of carbohydrate-binding sites prediction by StackCBPred on TS49 test set also indicates that StackCBPred predicts binding sites more accurately compared to the SPRINT-CBH predictor. Furthermore, the balanced accuracy measure indicates that the StackCBPred is not biased more towards the majority class rather it provides a balanced performance compared to SPRINT-CBH method.

Additionally, the performance of StackCBPred and SPRINT-CBH was further evaluated on the TS88 test set and their prediction results are listed in Table 8. Table 8 shows that the sensitivity of StackCBPred is 334.62% better than SPRINT-CBH. Besides, the miss rate of SPRINT-CBH is 0.870, which is very close to 1. Therefore, the specificity of the

Table 7
Comparisons of StackCBPred with SPRINT-CBH on a balanced and imbalanced independent test dataset, TS49.

Methods		Sens.	Spec.	BACC	ACC	MCC
SPRINT-CBH	Imbalanced	0.389	0.925	0.657	0.906	0.195
StackCBPred	Imbalanced	0.553	0.795	0.674	0.786	0.159
	Balanced	0.703	0.907	0.805	0.805	0.623

Table 8
Comparisons of StackCBPred with SPRINT-CBH on the imbalanced independent test dataset, TS88.

Methods	Sens.	Spec.	FOR	MR	BACC	MCC
SPRINT-CBH	0.130	0.997	0.003	0.870	0.564	0.257
StackCBPred	0.565	0.797	0.203	0.435	0.681	0.139

Table 9
Here, the contingency table is formed by comparing the predicted results of StackCBPred and SPRINT-CBH with actual class labels.

StackCBPred SPRINT-CBH	= Actual Class Label	≠ Actual Class Label
= Actual Class Label	21191	5057
≠ Actual Class Label	273	620

SPRINT-CBH is very high i.e., it most of the time predicts the sample point as the majority class (non-carbohydrate-binding) which results in low sensitivity. Additionally, the balanced accuracy of the SPRINT-CBH is 20.74% lower compared to StackCBPred. Thus, these results indicate that StackCBPred predicts a greater number of carbohydrate-binding and non-binding residues correctly and therefore is also a balanced predictor of carbohydrate-binding sites.

Moreover, Fig. 3 presents the ROC curves generated by StackCBPred and SPRINTCBH, while the predictions are evaluated on the imbalanced TS88 test set. The ROC curves show the TPR (sensitivity)/FPR (1-specificity) pairs at different classification thresholds. It is evident from the ROC curves that the StackCBPred provides higher TPR compared to SPRINT-CBH at different classification thresholds. Moreover, the AUC score given by StackCBPred is about 1.18% higher than that of SPRINT-CBH.

4. Discussions

In this section, we have first explained the statistical significance test performed on the TS88 independent test set and further we demonstrate the details of using the StackCBPred Software.

4.1. Statistical significance test

We performed McNemar's test on the TS88 independent test set to provide the statistical significance of our results. We could only perform statistical significance on TS88 test set as the prediction results from SPRINT-CBH web-server on TS49 test set do not match the results mentioned in the paper. The differences in the accuracies that are obtained from SPRINT-CBH web-server and the paper could be an outcome of using TS49 test set for training the SPRINT-CBH web-server model. At first, we set our null and alternative hypotheses. For the null hypothesis, we assume that there is no difference between StackCBPred and SPRINT-CBH predictors whereas, for the alternate hypothesis, we assume that there is a significant difference between StackCBPred and SPRINT-CBH predictors. Then, we prepare two different contingency tables and conduct McNemar's test separately. Finally, depending upon the p-value obtained from the McNemar's test, we either accept or reject our null hypothesis. The detailed approach is shown below:

- Null Hypothesis: There is no difference between StackCBPred and SPRINT-CBH predictors.
- Alternate Hypothesis: There is a significant difference between StackCBPred and SPRINTCBH predictors.

Construction of two different contingency tables:

To perform McNemar's test, we set alpha value of 0.05 as the cutoff for significance test and run McNemar's test on both the contingency tables are shown in Table 9 and Table 10. The McNemar's test for two different contingency tables above resulted in a p-value of < 0.01 which, is less than 0.05. Therefore, we reject the null hypothesis and accept the alternative hypothesis that there is a significant difference between StackCBPred and SPRINT-CBH.

4.2. StackCBPred online server

StackCBPred is available as an online server at http://bmll.cs.uno.edu/add. The details of using StackCBPred are as follows: StackCBPred accepts input as a single protein sequence and generates an output that contains the predicted annotation for every residue in the input sequence as 1/0 (1 indicates binding and 0 indicate non-binding) followed by nonbinding probability and binding probability. A screenshot of the top page of StackCBPred online server with example sequence is shown in Fig. 4. To use StackCBPred, users need to provide a job title, protein sequence, and passkey before submitting a job to StackCBPred server. An example job title and protein sequence are also provided for

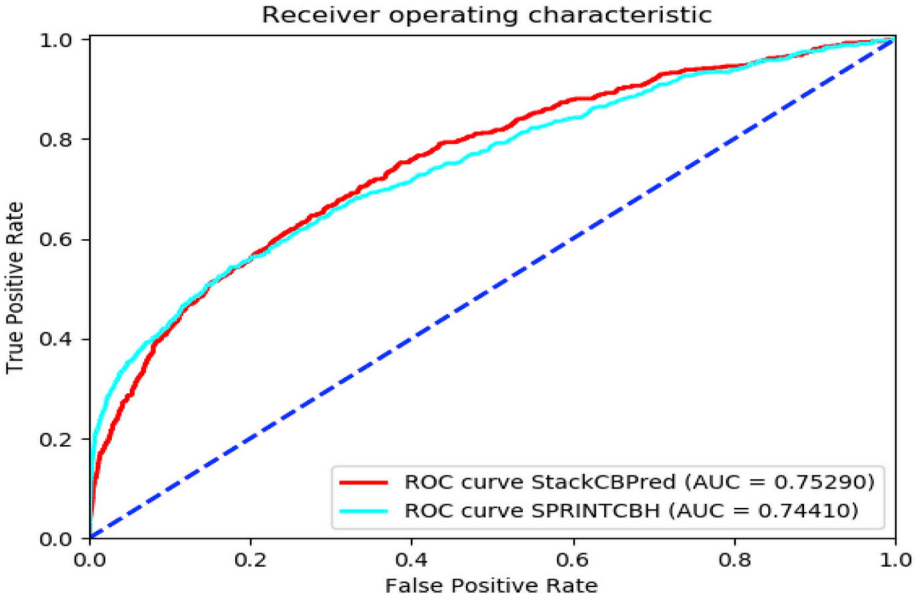


Fig. 3. Comparison of ROC and AUC scores given by StackCBPred and SPRINT-CBH on the imbalanced independent test dataset, TS88.

Table 10
Here, the contingency table is formed by comparing the predicted results of StackCBPred and SPRINT-CBH with each other.

StackCBPred SPRINT-CBH	Carbohydrate-Binding	Non-Carbohydrate-Binding	
Carbohydrate-Binding	485	48	533 (1.96%)
Non-Carbohydrate-Binding	5282	21326	26608 (98.03%)
	5767 (21.24%)	21374 (78.7%)	27141

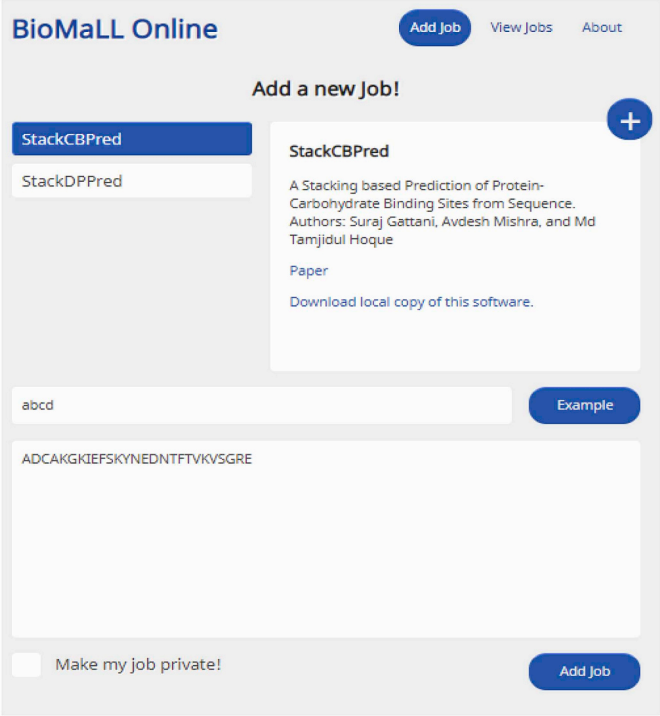


Fig. 4. Screenshot shows the top page of the StackCBPred online server with an example sequence.

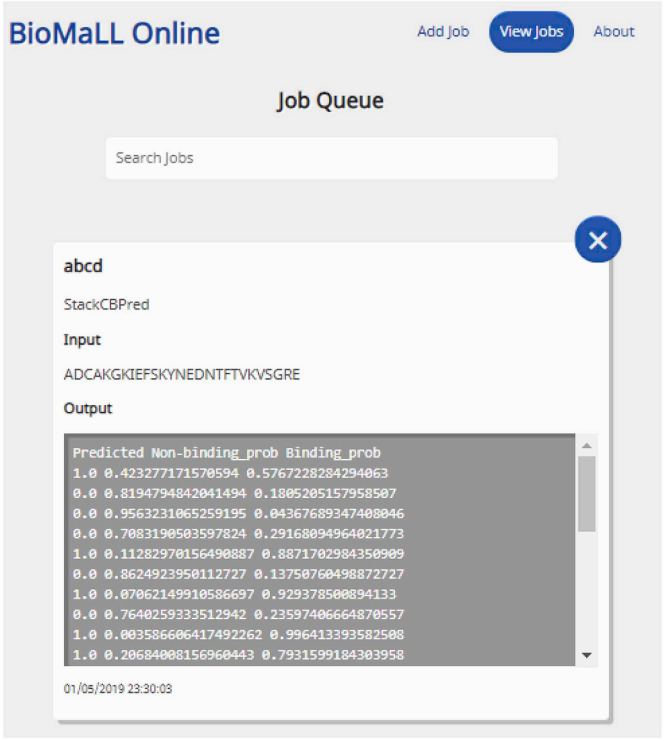


Fig. 5. Screenshot shows the output page of the StackCBPred online server for an example sequence.

the users which are populated in the respective text boxes by clicking on the "Example" menu. The passkey is optional and is used to make the output of the job either private or public. By default, the outputs of the jobs submitted to StackCBPred are public, but the users can make the job private by entering the passkey before submitting a job to StackCBPred server. The screenshot of the output of the StackCBPred online server is shown in Fig. 5.

5. Conclusions

In this work, we have developed a Stacking-based machine learning predictor, named StackCBPred, for the prediction of protein-carbohydrate binding sites directly from the protein sequence. We collected a benchmark dataset and two independent test datasets of high-resolution carbohydrate-binding proteins to train, validate and independently test StackCBPred. Several important evolution-derived, sequence-based and structural features were extracted and chosen in an incremental fashion to find the trained best performing model. In addition, an advanced machine learning technique called stacking was implemented to ensure robust performance. We used incrementally chosen features to train the ensemble of predictors at the first-stage (i.e., base-layer). Then, we combined the output from the base-learners with the original features and used it as an input to the predictor at second-stage (i.e., meta-layer). Eventually, the meta-layer predictor of the StackCBPred achieves a 10-fold CV balanced accuracy and sensitivity of 86.00% and 86.09% respectively, on a balanced benchmark dataset. For the balanced independent test dataset, TS49, StackCBPred attains a balanced accuracy and sensitivity of 80.51% and 70.28%, respectively. Furthermore, for the new imbalanced independent test dataset TS88 introduced in this work, StackCBPred attains a balanced accuracy and sensitivity of 68.46% and 56.39%, respectively. These results allow us to conclude that the stacking technique helps improve the accuracy significantly by reducing the generalization error. Moreover, comparative results highlight that the proposed method, StackCBPred, outperforms the existing method based on both benchmark and independent test datasets. These outcomes help us surmise that the StackCBPred can be effectively used for the rapid annotation of carbohydrate-binding sites directly from the sequence and can provide insight in treating critical diseases.

Data availability

The data of this research and software code oof the tool can be found here: http://cs.uno.edu/~tamjid/Software/StackCBPred/code_data.zip.

Funding

The authors gratefully acknowledge the Louisiana Board of Regents through the Board of Regents Support Fund LEQSF (2016-19)-RD-B-07.

Declaration of competing interest

There is no conflict of interest with any of the authors or the suggested reviewers.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.carres.2019.107857.

References

[1] C. Shionyu-Mitsuyama, et al., An empirical approach for structure-based prediction of carbohydrate-binding sites on proteins, Protein Eng. 16 (7) (2003) 467–478.
[2] M.d.C. Fernandez-Alonso, et al., Protein-carbohydrate interactions studied by NMR: from molecular recognition to drug design, Curr. Protein Pept. Sci. 13 (2012) 816–830.
[3] N. Sharon, H. Lis, Lectins, 2 ed., Springer, The Netherlands, 2003, p. 454.
[4] I. Shin, S. Park, M.r. Lee, Carbohydrate microarrays: an advanced Technology for functional studies of glycans, Chem. Eur J. 11 (2005) 2894–2901.
[5] M. Wimmerová, et al., Stacking interactions between carbohydrate and protein quantified by combination of theoretical and experimental methods, PLoS One 7 (10) (2012) e46032.
[6] G. Taherzadeh, et al., Sequence-based prediction of protein − carbohydrate binding sites using support vector machines, J. Chem. Inf. Model. 56 (2016).
[7] M.P. McKinley, et al., Human Anatomy, fourth ed. ed, McGraw-Hill Education, New York, NY, 2015.
[8] A. Malik, S. Ahmad, Sequence and structural features of carbohydrate binding in proteins and assessment of predictability using a neural network, BMC Struct. Biol. 7 (1) (2007).
[9] A. Brown, M.K. Higgins, Carbohydrate binding molecules in malaria pathology, Curr. Opin. Struct. Biol. 20 (5) (2010) 560–566.
[10] K. François, J. Balzarini, Potential of carbohydrate-binding agents as therapeutics against enveloped viruses, Med. Res. Rev. 32 (2012) 349–387.
[11] A. Raz, S. Nakahara, Biological modulation by lectins and their ligands in tumor progression and metastasis, Anti Cancer Agents Med. Chem. 8 (1) (2008) 22–36.
[12] C. Taroni, S. Jones, J.M. Thornton, Analysis and prediction of carbohydrate binding sites, Protein Eng. 13 (2) (2000) 89–98.
[13] H. Nassif, et al., Prediction of protein-glucose binding sites using support vector machines, Proteins: Struct. Funct. Bioinform. 77 (1) (2009) 121–132.
[14] K.-C. Tsai, et al., Prediction of carbohydrate binding sites on protein surfaces with 3-dimensional probability density distributions of interacting atoms, PLoS One 7 (7) (2012) e40846.
[15] M.M. Gromiha, K. Veluraja, K. Fukui, Identification and analysis of binding site residues in proteincarbohydrate complexes using energy based approach, Protein Pept. Lett. 21 (8) (2014) 799–807.
[16] N.R.S. Shanmugam, et al., Identification and analysis of key residues involved in folding and binding of protein-carbohydrate complexes, Protein Pept. Lett. 25 (4) (2018) 379–389.
[17] L. Deng, et al., Boosting prediction performance of protein–protein interaction hot spots by using structural neighborhood properties, J. Comput. Biol. 20 (11) (2013) 878–891.
[18] C. Lei, J. Ruan, A novel link prediction algorithm for reconstructing protein–protein interaction networks by topological similarity, Bioinformatics 29 (3) (2013) 355–364.
[19] S. Liang, et al., Protein binding site prediction using an empirical scoring function, Nucleic Acids Res. 34 (13) (2006) 3698–3707.
[20] V.S. Rao, et al., Protein-protein interaction detection: methods and analysis, Int. J. Proteomics 2014 (2014).
[21] S. Iqbal, M.T. Hoque, PBRpredict-Suite: a suite of models to predict peptide-recognition domain residues from protein sequence, Bioinformatics (2018) bty352-bty352.
[22] A. Lavi, et al., Detection of peptide-binding sites on protein surfaces: the first step towards the modeling and targeting of peptide-mediated interactions, Proteins: Struct. Funct. Bioinf. 81 (12) (2013) 2096–2105.
[23] E. Petsalaki, et al., Accurate prediction of peptide binding sites on protein surfaces, PLoS Comput. Biol. 5 (3) (2009).
[24] G. Taherzadeh, et al., Sequence-based prediction of protein–peptide binding sites using support vector machine, J. Comput. Chem. 37 (13) (2016) 1223–1229.
[25] C.-K. Lin, C.-Y. Chen, PiDNA: predicting protein–DNA interactions with structural models, Nucleic Acids Res. 41 (2013) W523–W530.
[26] A. Mishra, P. Pokhrel, M.T. Hoque, StackDPPred: a stacking based prediction of DNA-binding protein from sequence, Bioinformatics (2018) bty653.
[27] J. Si, et al., MetaDBSite: a meta approach to improve protein DNA-binding sites prediction, BMC Syst. Biol. 5 (2011).
[28] H. Zhao, et al., Predicting DNA-binding proteins and binding residues by complex structure prediction and application to human proteome, PLoS One 9 (2014).
[29] Y. Murakami, et al., PiRaNhA: a server for the computational prediction of RNA-binding residues in protein sequences, Nucleic Acids Res. 38 (2010) W412–W416.
[30] T. Zhang, et al., Analysis and prediction of RNA-binding residues using sequence, evolutionary conservation, and predicted secondary structure and solvent accessibility, Curr. Protein Pept. Sci. 11 (7) (2010) 609–628.
[31] X. Zhang, S. Liu, RBPPred: predicting RNA-binding proteins from sequence using SVM, Bioinformatics 33 (6) (2017) 854–862.
[32] H. Zhao, Y. Yang, Y. Zhou, Highly accurate and high-resolution function prediction of RNA binding proteins by fold recognition and binding affinity prediction, RNA Biol. 8 (6) (2011) 988–996.
[33] A. Bolia, Z.N. Gerek, S.B. Ozkan, BP-Dock: a flexible docking scheme for exploring protein–ligand interactions based on unbound structures, J. Chem. Inf. Model. 54 (3) (2014) 913–925.
[34] Y. Komiyama, et al., Automatic generation of bioinformatics tools for predicting protein–ligand binding sites, Bioinformatics 32 (6) (2016) 901–907.
[35] T. Litfin, Y. Zhou, Y. Yang, SPOT-ligand 2: improving structure-based virtual screening by binding-homology search on an expanded structural template library, Bioinformatics 33 (8) (2017) 1238–1240.
[36] Y. Yang, J. Zhan, Y. Zhou, SPOT-Ligand: fast and effective structure-based virtual screening by binding homology search according to ligand and receptor similarity, J. Comput. Chem. 37 (18) (2016) 1734–1739.
[37] S.-J. Yen, Y.-S. Lee, Under-Sampling Approaches for Improving Prediction of the Minority Class in an Imbalanced Dataset, Springer Berlin Heidelberg, 2006, pp. 731–740.
[38] A. Malik, et al., PROCARB: a database of known and modelled carbohydrate-binding protein structures with sequence-based prediction tools, Adv. Bioinform. 2010 (2010) 436036.
[39] M. Johnson, et al., NCBI BLAST: a better web interface, Nucleic Acids Res. 36 (2008) W5–W9.
[40] A.K. Biswas, N. Noman, A.R. Sikder, Machine learning approach to predict protein phosphorylation sites by incorporating evolutionary information, BMC Bioinf. 11 (273) (2010).
[41] N. Islam, et al., A balanced secondary structure predictor, J. Theor. Biol. 389 (2016) 60–71.
[42] S. Iqbal, A. Mishra, T. Hoque, Improved prediction of accessible surface area results in efficient energy function application, J. Theor. Biol. 380 (2015) 380–391.
[43] R. Verma, G.C. Varshney, G.P.S. Raghava, Prediction of mitochondrial proteins of malaria parasite using split amino acid composition and PSSM profile, Amino Acids 39 (1) (2010) 101–110.
[44] F. Glaser, et al., ConSurf: identification of functional regions in proteins by surface-mapping of phylogenetic information, Bioinformatics 19 (1) (2003) 163–164.
[45] S.F. Altschul, et al., Basic local alignment search tool, J. Mol. Biol. 215 (1990) 403–410.
[46] H. Saini, et al., Protein structural class prediction via k-separated Bigrams using position specific scoring matrix, J. Adv. Comput. Intell. Intell. Inform. 18 (2014) 474–479.
[47] A. Sharma, et al., A feature extraction technique using bi-gram probabilities of position specific scoring matrix for protein fold recognition, J. Theor. Biol. 320 (2013) 41–46.
[48] E. Faraggi, et al., SPINE X: improving protein secondary structure prediction by multi-step learning coupled with prediction of solvent accessible surface area and backbone torsion angles, J. Comput. Chem. 33 (3) (2012) 259–267.
[49] T. Hamelryck, An amino acid has two sides: a new 2D measure provides a different view of solvent exposure, Proteins: Structure, Function, Bioinformatics 59 (1) (2005) 38–48.
[50] R. Heffernan, et al., Highly accurate sequence-based prediction of half-sphere exposures of amino acid residues in proteins, Bioinformatics 32 (6) (2016) 843–849.
[51] R. Heffernan, K. Paliwal, J. Lyons, J. Singh, Y. Yang, Y. Zhou, Single-sequence-based prediction of protein secondary structures and solvent accessibility by deep whole-sequence learning, J. Comput. Chem. 39 (26) (2018) 2210–2216.
[52] R. Heffernan, et al., Capturing non-local interactions by long short term memory bidirectional recurrent neural networks for improving prediction of protein secondary structure, backbone angles, contact numbers, and solvent accessibility, Bioinformatics 33 (18) (2017) 2842–2849.
[53] J. Meiler, et al., Generation and evaluation of dimension-reduced amino acid parameter representations by artificial neural networks, J. Mol. Model. 7 (2001) 360–369.
[54] A. Bah, J.D. Forman-Kay, Modulation of intrinsically disordered protein function by post-translational modifications, J. Biol. Chem. 291 (2016) 6696–6705.
[55] Y.-H. Lina, et al., The intrinsically disordered N-terminal domain of galectin-3 dynamically mediates multisite self-association of the protein through fuzzy interactions, J. Biol. Chem. 292 (2017) 17845–17856.
[56] R. Sharma, et al., MoRFPred-plus: computational identification of MoRFs in protein sequences using physicochemical properties and HMM profiles, J. Theor. Biol. 437 (2017).
[57] R. Sharma, et al., OPAL: prediction of MoRF regions in intrinsically disordered protein sequences, Bioinformatics 34 (11) (2018) 1850–1858.
[58] R. Sharma, et al., OPAL+: Length-specific MoRF Prediction in Intrinsically Disordered Protein Sequences. PROTEOMICS, (2018), p. 1800058.
[59] T. Hastie, R. Tibshirani, J. Friedman, The elements of statistical learning, Springer Series in Statics, 2 ed., Springer-Verlag, New York, 2009.
[60] D.H. Wolpert, Stacked generalization, Neural Netw. 5 (2) (1992) 241–259.
[61] Q. Hu, et al., A stacking-based approach to identify translated upstream open reading frames in Arabidopsis Thaliana, Bioinformatics Research and Applications International Symposium on Bioinformatics Research and Applications, 2015, pp. 138–149.
[62] S. Nagi, D.K. Bhattacharyya, Classification of microarray cancer data using ensemble approach, Netw. Model. Anal. Health Inf. Bioinf. 2 (3) (2013) 159–173.
[63] A. Verma, S. Mehta, A comparative study of ensemble learning methods for classification in bioinformatics, 7th International Conference on Cloud Computing, Data Science & Engineering - Confluence, IEEE, Noida, India, 2017.
[64] V.N. Vapnik, An overview of statistical learning theory, IEEE Trans. Neural Netw. 10 (5) (1999).
[65] J.H. Friedman, Greedy function approximation: a gradient boosting machine, Ann. Stat. 29 (5) (2001) 1189–1232.
[66] L. Breiman, Bagging predictors, Mach. Learn. 24 (2) (1996) 123–140.
[67] P. Geurts, D. Ernst, L. Wehenkel, Extremely randomized trees, Mach. Learn. 63 (1) (2006) 3–42.

[68] T.K. Ho, Random decision forests, in document analysis and recognition, Proceedings of the Third International Conference on. 1995, IEEE, Montreal, Que., Canada, 1995, pp. 278–282.
[69] N.S. Altman, An introduction to kernel and nearest-neighbor nonparametric regression, Am. Stat. 46 (1992) 175–185.
[70] A. Szilágyi, J. Skolnick, Efficient prediction of nucleic acid binding function from low-resolution protein structures, J. Mol. Biol. 358 (3) (2006) 922–933.
[71] T. Chen, C. Guestrin, XGBoost: a scalable tree boosting system, Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2016, pp. 785–794.
[72] F. Pedregosa, et al., Scikit-learn: machine learning in python, J. Mach. Learn. Res. 12 (2012).
[73] J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, J. Mach. Learn. Res. 13 (2012) 281–305 3/1/2012.
[74] C. Lin, et al., LibD3C: ensemble classifiers with a clustering and dynamic selection strategy, Neurocomputing 123 (2014) 424–435.
[75] H. Peng, F. Long, C. Ding, Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy, IEEE Trans. Pattern Anal. Mach. Intell. 27 (2005) 1226–1238.
[76] Z. Quan, et al., A novel features ranking metric with application to scalable visual and bioinformatics data classification, Neurocomputing 173 (2016) 346–354.

Suraj Gattani received a Master's in Computer Science from the University of New Orleans in Spring 2019. He is currently working as a Machine Learning Engineer at GV20 Therapeutics, Cambridge, MA. His interests are Machine Learning, Bioinformatics, and Robotics. His objective is to use machine learning methods to solve real-world problems.

Avdesh Mishra received an M.S. and a Ph.D. degree in Computer Science from the University of New Orleans, New Orleans, LA, USA, in 2015 and 2019, respectively. He is currently an Assistant Professor with the Electrical Engineering and Computer Science Department, Texas A& M University-Kingsville, Kingsville, TX, USA. Before starting his M.S. degree, he worked as a software engineer in software industry, BitsCrafters. He is interested in Bioinformatics, Artificial Intelligence, Machine Learning, Data Science and Big Data research.

Md Tamjidul Hoque received a Ph.D. degree in Information Technology from Monash University, Melbourne, VIC, Australia, in 2008. He is currently an Associate Professor with the Computer Science Department, University of New Orleans, New Orleans, LA, USA. From 2011 to 2012, he was a Post-Doctoral Fellow with Indiana University–Purdue University Indianapolis, Indianapolis, IN, USA. From 2007 to 2011, he was a Research Fellow with Griffith University, Brisbane, QLD, Australia. His current research interests include machine learning, evolutionary computation, and artificial intelligence, applying toward hard optimization problems especially for bioinformatics problems such as protein structure-prediction, disorder predictor, and energy function.