Prediction of protein-carbohydrate binding sites from protein primary sequence

Quazi Farah Nawar[a,c], Md Muhaiminul Islam Nafi[a,c], Tasnim Nishat Islam[b], M Saifur Rahman[a,d]

[a]*Department of CSE, BUET, Dhaka 1000, Bangladesh*
[b]*Department of EEE, BUET, Dhaka 1000, Bangladesh*
[c]*Equal contribution*
[d]*Corresponding author*

Abstract

A protein is a large complex macromolecule that has a crucial role in performing most of the work in cells and tissues. It is made up of one or more long chains of amino acid residues. Another important biomolecule, after DNA and protein, is carbohydrate. Carbohydrates interact with proteins to run various biological processes. Several biochemical experiments exist to learn the protein-carbohydrate interactions, but they are expensive, time consuming and challenging. Therefore developing computational techniques for effectively predicting protein-carbohydrate binding interactions from protein primary sequence has given rise to a prominent new field of research. In this study, we propose *StackCBEmbed*, an ensemble machine learning model to effectively classify protein-carbohydrate binding interactions at residue level. StackCBEmbed combines traditional sequence-based features along with features derived from a pre-trained transformer-based protein language model. To the best of our knowledge, ours is the first attempt to apply protein language model in predicting protein-carbohydrate binding interactions. StackCBEmbed achieved sensitivity, specificity and balanced accuracy scores of 0.730, 0.821, 0.776 and 0.666, 0.818, 0.742 in two separate independent test sets. This performance is superior compared to the earlier prediction models benchmarked in the same datasets. We thus hope that StackCBEmbed will discover novel protein-carbohydrate interactions and help advance the related fields of research. StackCBEmbed is freely available as python scripts at `https://github.com/nafiislam/StackCBEmbed`.

1. Introduction

Nucleic acids, proteins, carbohydrates and lipids are important biomolecules for any organism. Among these, carbohydrates come right after DNA and proteins in terms of importance [1]. Carbohydrates interact with protein molecules to facilitate various biological processes, such as, cellular adhesion, cellular recognition, protein folding, ligand recognition, and so on [2, 3, 4]. Moreover, they provide protection of human cells against pathogens [5] as well as play an important role as biomarkers or drug targets [6, 7, 8, 9]. In order to identify protein-carbohydrate interactions, several experimental techniques have been developed. These include X-ray crystallography, nuclear magnetic resonance (NMR) spectroscopy study, molecular modeling, dual polarization interferometry and so on. However, weak binding affinity and synthetic complexity of individual carbohydrates have made these techniques more expensive, time-consuming, and challenging [10]. Therefore, developing a computational technique for effectively predicting protein-carbohydrate binding sites has become an urgent necessity. The computational approaches concentrate on locating the sites of proteins that bind to carbohydrates.

Several computational methods can be found in the literature that leverage structural features to locate protein-carbohydrate binding sites. Taroni et al. [11] first proposed a method which used known protein structure to predict protein-sugar binding sites. They utilized six different attributes of each site, such as, solvation potential, residue propensity, hydrophobicity, planarity, protrusion and relative accessible surface area. Through a feature selection process three of these attributes (residue propensity, protrusion, and relative accessible surface area) were finally selected. These three parameters were then used to calculate the probability for a surface patch to be a carbohydrate binding site. This method provided 65% accuracy for 40 protein-carbohydrate sequences.

A method called COTRAN, proposed by Sujatha and Balaji [12], is able to predict galactose-binding sites from known galactose-binding proteins. Galactose-binding proteins are a special class of proteins, and galactose binding sites vary across different protein-galactose complexes. No common features are known to be shared by these proteins to specifically bind to their targets. This study assumed that common rules exist for how they recognize galactose. As a result, the study examines 18 protein-galactose 3D structures from 7 different families to identify a unique pattern for galactose binding sites. Each family has similar binding sites, but these sites differ between families.

Another method called InCa-SiteFinder was proposed by Kulharia et al. [13] to predict inositol and carbohydrate binding sites on the protein surface. This method utilizes amino acid propensities and van der waals interaction energy between protein and a probe. In order to predict glucose binding sites, Nassif et al. [14] utilized several chemical and residue based features such as charges, hydrophobicity and hydrogen bonding with support vector machine (SVM) as the classifier. Tsai et al. [15] fed probability distributions of interacting atoms in protein surfaces to neural networks and SVMs to predict the binding sites. An energy based approach was proposed in [16] to identify the protein-carbohydrate binding sites and the residues which have significant roles in binding. This method distinguished 3.3% of residues as binding sites in protein-carbohydrate complexes. In this study, the dominance of Tryptophan (TRP) to communicate with carbohydrates through aromatic-aromatic interactions had also been observed by performing the binding propensity analysis.

Shanmugam et al. proposed a method to identify and study the residues, which are involved in both folding and binding of protein-carbohydrate complexes [17]. They identified the stabilizing residues by means of hydrophobicity, long range interactions and conservations. The binding site residues were identified using a a distance cutoff of 3.5Å between any heavy atoms in protein and ligand. The residues which were common in stabilizing and binding, are termed as key residues. The study found that most of the key residues are present in β-strands. All-α, $\alpha + \beta$ and α/β structural classes have more key residues than the other protein classes. Nevertheless, the structure-based methods discussed thus far rely on protein structures that are not always available, thereby rendering those ineffective as well as highlighting the importance of sequence-based approaches.

Malik and Ahmed [6] proposed the first sequence-based computational technique in which evolutionary features from the Position Specific Scoring Matrix (PSSM) were used to train a neural network that achieved 87% sensitivity and 23% specificity in a test set of 40 protein-carbohydrate complexes. Pai et al. [18] and Agarwal et al. [19] proposed similar PSSM based methods to predict mannose-binding sites. Another method, SPRTNT-CBH was proposed in [10] where PSSM features along with other predicted structural features (referred to as sequence-derived structural information) were employed to train an SVM classifier. The classifier achieved 22.3% sensitivity and 98.8% specificity on an independent test set of 50 protein-carbohydrate complexes. Both of the methods mentioned above have issues with imbalanced prediction results, either having high sensitivity and low specificity, or vice verse.

To address these concerns, StackCBPred [20] employed a stacking ensemble classifier to enhance the prediction accuracy in terms of sensitivity (56.5%) and specificity (79.7%). This classifier was

trained using a balanced training dataset and incorporated various evolutionary features as well as sequence-based predicted structural features. This model managed to somewhat mitigate the prediction bias observed in previous methods. However, there is still potential for further improvement in prediction performance. By incorporating more relevant features and exploring various other machine learning (ML) and deep learning (DL) architectures, the prediction performance could be significantly enhanced.

In this study, we have introduced a stacking ensemble prediction model named StackCBEmbed. This model seamlessly integrates features directly extracted from protein sequences with attributes derived from a pre-trained transformer-based protein language model. An increasingly successful trend in various protein attribute prediction problems has been the utilization of recently developed protein language models [21, 22]. These pre-trained models enable the generation of informative features for individual proteins, as well as the residues therein, offering enhanced predictive power while being computationally less intensive compared to evolutionary and other features that necessitate time-consuming database searches. To the best of our knowledge, ours is the first work to incorporate protein embedding features from a language model in the context of protein-carbohydrate binding sites prediction. Our method showed a significant improvement in terms of several prediction metrics, outperforming the state-of-the-art carbohydrate binding site prediction methods. The key contributions of our work can be summarized as follows.

- We propose StackCBEmbed, a stacking ensemble prediction model protein-carbohydrate binding sites. The architecture of the ensemble is determined through an objective approach, searching from a set of base predictors.
- We have investigated features from a transformer-based protein language model (ProtT5-XL-U50) alongside conventional sequence-based and evolutionary features. Our results demonstrate the superiority of the embeddings extracted from the protein language model compared to the other features.
- As the dataset is heavily imbalanced, we have applied an ensemble random undersampling approach for data balancing.
- StackCBEmbed outperforms state-of-the-art methods for predicting protein-carbohydrate binding sites, demonstrating a notable improvement across a number of prediction metrics.

2. Materials and Methods

2.1. Dataset

Zhao et al. [23] extracted 604 protein-carbohydrate complex structures from PROCARB database [24]. After removing glycoproteins, they selected only those proteins with more than five carbohydrate-binding residues (i.e., residues which have one or more heavy atoms that are within 4.5 Å of any heavy atom of carbohydrates). They further removed the redundant proteins using BLASTCLUST [25] with a sequence identity cutoff of 30%. They thus obtained 113 protein-carbohydrate complexes. Taherzadeh et al. [10] used the Glyco3d database (`http://glyco3d.cermav.cnrs.fr/home.php`) to obtain more than 1000 three-dimensional structures of protein-carbohydrate complexes. After combining these two sets, they removed low-resolution structures (>3 Å) and homologous proteins (30% sequence identity cut-off). They thus obtained 152 protein-carbohydrate complexes. Out of these, 50 proteins were randomly chosen and separated out for independent test set (TS50), and the remaining set (102 proteins), labeled as TR102, was used for model training in their study. Gattani et al. [20] discarded 2 sequences from TR102, and 1 sequence from TS50

as those contained non-standard amino acids. They referred to the updated sets as the benchmark dataset and TS49 respectively. In our study we have used the benchmark dataset for training our prediction model and TS49 for independent testing. However, we had to leave out two more sequences (IDs: 2c3hB and 2vngB) from the benchmark dataset due to discrepancy in the number of residues in the the FASTA files vs. the PSSM files that we generated thereof.

Gattani et al. created a second independent test set from PROCARB as follows. From the PROCARB604 dataset, the proteins whose ID's matched the protein ID's that were present in the benchmark or TS49 dataset were removed. Redundant proteins with a sequence identity cutoff of 30% were then removed. They thus obtained a set of 88 sequences, which was named TS88. Like them, we too have used this as a second independent test set in our study.

The benchmark dataset contains 98 sequences with 1014 binding residues and 25705 non-binding residues. We randomly selected 30% of these residues to create a validation set, consisting of 275 binding residues and 7741 non-binding residues. The remaining portion was designated as the training set. The binding and non-binding residue distributions of the datasets used in our study are listed in Table 1, from which it is evident that all the datasets are highly imbalanced.

Table 1: Number of binding and non-binding sites in the datasets.

Dataset Type	**Count Of**		
	Proteins	**Binding sites**	**Non-binding sites**
Training set	98	739	17964
Validation set		275	7741
TS49	49	508	13230
TS88	88	688	26453

2.2. Feature Extraction

Feature extraction is a very important step in the prediction pipeline. In this study, we utilized two types of features: the traditional sequence-based features and the embeddings derived from pre-trained transformer-based protein language models. These features are described below. Notably, after featurization, Yeo–Johnson transformation [26] was applied, followed by standardization, using *PowerTransformer* from *scikit-learn* python package.

i **Position specific scoring matrix (PSSM)**
Position specific scoring matrix (PSSM) expresses the evolution-derived information in protein sequences. Evolutionarily conserved residues have occupied a prominent place in several biological analysis [27, 28, 29, 30, 31, 32] and they have considerably important functional roles in binding. We generated PSSM using PSI-BLAST [33] with E-value = 0.001, with 5 iterations. The PSSM is a $L \times 20$ dimensional matrix (L is the length of the protein sequence) which can be represented by :

$$PSSM = \begin{bmatrix} P_{1,1} & P_{1,2} & \dots & P_{1,20} \\ \vdots & \ddots & \vdots & \vdots \\ P_{L,1} & P_{L,2} & \dots & P_{L,20} \end{bmatrix}$$

Here, $P_{i,j}$ indicates the score of the amino acid at position i being changed to amino acid j during the evolutionary process. For each residue we thus get a 20 dimensional feature vector.

ii **Monogram (MG)**
Monogram (MG) [34, 35] feature for a residue is extracted from PSSM by summing up the scores

for that residue along the protein sequence's length. Thus it results in a feature vector of length 1 for each residue and is location invariant.

$$M_m = \sum_{i=1}^{L} P_{i,m} \tag{1}$$

where, $1 <= m <= 20$ and L is the length of the sequence.

iii **Di-peptide composition (DPC)**
Di-peptide composition (DPC) [36] can also be computed from PSSM to capture the partial sequence-order information. As, there are 20×20 combinations of the dipeptides, this feature can be represented as a 20×20 matrix $DPC[i, j], 1 \leq i, j \leq 20$, which can later be flattened into a 400-dimensional vector. For the residue at position $k, 1 \leq k \leq L$, $DPC[i, j]$ is calculated as follows.

$$DPC[i, j] = P_{k,i} \cdot P_{k+1,j} + P_{k,i} \cdot P_{k-1,j} \tag{2}$$

Note that for the first and last residues, the indices $k-1$ and $k+1$ respectively are out of range, therefore the corresponding P values are assumed to be 0. In other words, for the terminal residues, only one of the two terms in the sum has contribution.

iv **Accessible Surface Area (ASA)**
In order to predict the structural and functional features of proteins as well as the sequence-structure-function relationship, the knowledge of solvent exposure is important. Accessible Surface Area (ASA) [37] measures the solvent accessibility of a biomolecule and can play a crucial role in predicting binding sites. Here SPIDER3 program [38, 39, 40] was used to predict ASA probabilities at residue level directly from each of the protein sequence.

v **Half Sphere Exposure (HSE)**
ASA has several drawbacks, however. For example it is quite difficult to determine to what extent a residue is buried and also it is impossible to differentiate the partial buried residue and deeply buried residue through ASA. In order to solve these issues, HSE had been introduced [41]. HSE seperates a residue's sphere into two half spheres : HSE-up and HSE-down. This measure has several interesting properties in characterization of the residue's spatial neighborhood as well as it does not rely on a full-atom model. Thus it is very convenient to apply this measure in protein struture modeling and prediction analysis. HSE-up and HSE-down were also extracted using the SPIDER3 program.

vi **Torsion angles (TA)**
The torsion angles Phi (ϕ) and Psi (ψ) determine the shape of the protein backbone structure. The predicted values of these two angles have been used as features as they have crucial roles in analyzing protein-carbohydrate interactions. Here too, we have used SPIDER3 for the predicted torsion angles.

vii **Physicochemical properties (PHYSICO)**
For each of the 20 standard amino acids, we utilized seven different physicochemical properties, as utilized in [42]: steric parameter, polarizability, normalized Van der Waals volume, hydrophobicity, isoelectric point, helix probability and sheet probability.

viii **Embedding from pre-trained protein language model (EPLM)**
From the latest Nàtural Language Processing (NLP) research [22], transformer models were trained on large protein corpora and these are known as protein language models. The output from the last layer of these models can be used as features in various prediction problems. In our study, six of the models from [22] were used – ProtBert, ProtBert-BFD, ProtT5-XL-BFD, ProtT5-XL-U50, ProtAlbert and ProtXLNet. ProtBert and ProtBert-BFD are BERT based models [43] trained on UniRef100 and BFD-100 corpora respectively. ProtAlbert model was acqiured from Albert model [44] while ProtXLNet was acquired from XLNet model [45]. Both were trained on the UniRef100 corpora. On the otherhand, ProtT5-XL-BFD was obtained from the T5 model [46] trained on BFD corpora. It was then fine tuned on the UniRef50 dataset to create the ProtT5-XL-U50 model. Upon comparative analysis of the embeddings for our problem (see Results section), we decided to go with the ProtT5-XL-U50 embedding, which is a 1024 dimensional vector.

2.3. Performance Evaluation Criteria

We have used several well-established metrics [47, 48] to objectively measure the performance of our prediction model. These are sensitivity (SN), specificity (SP), accuracy (Acc), balanced accuracy (BACC), Precision (Prec), F1-score (F1) and Matthew's Correlation Coefficient (MCC). These can be mathematically defined as follows:

$$SN = \frac{TP}{TP + FN} \tag{3}$$

$$SP = \frac{TN}{TN + FP} \tag{4}$$

$$Acc = \frac{TP + TN}{FP + TP + TN + FN} \tag{5}$$

$$BACC = \frac{SN + SP}{2} \tag{6}$$

$$Prec = \frac{TP}{FP + TP} \tag{7}$$

$$F1 = \frac{2TP}{2TP + FP + FN} \tag{8}$$

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FN) \times (TP + FP) \times (TN + FP) \times (TN + FN)}} \tag{9}$$

Here, TP, FP, TN, and FN stand for True Positive, False Positive, True Negative, and False Negative respectively. Additionally, we also assessed the area under receiver operating characteristic curve (AUROC or AUC in short) [49] and area under precision-recall curve (auPR in short) [50].

2.4. Feature selection

Using the aforementioned feature descriptors, we used incremental feature selection (IFS) to obtain the optimal combination of features. We took a phased approach to IFS (Algorithm 1), where at each phase, we try to augment the current set of features with one additional feature group. Among all the groups, the augmentation that yields the best performance is chosen and the feature set is updated. The updated feature set is used in the next phase and the process repeats. Thus, while the search space of feature group combinations is exponential, only a limited set of combinations is explored in a greedy manner. Finally, of all the combinations explored, the one with the best performance is chosen. To measure the performance of each combination, we trained a classification model with those features using Extreme Gradient Boosting (XGB) from the scikit-learn package in Python [51], with default parameters. The benchmark dataset, balanced using random undersampling, was used. We used XGB since it has been very successful in numerous ML challenges in Kaggle (`https://www.kaggle.com/`).

Algorithm 1: Iterative Feature Selection

Data: Input feature set X
Result: Selected feature set Y

```
Y ← ∅
S ← ∅
for i ← 1 to X.length do
    A ← ∅
    foreach feature group f in X − S do
        B ← S ∪ f
        Train XGB on features in B using 10-fold CV
        if F1-score(B) > F1-score(A) then
            A ← B
    S ← A
    if F1-score(A) > F1-score(Y) then
        Y ← A
return Y;
```

2.5. Framework of proposed method

In order to improve the prediction performance, we implemented a stacking ensemble model [52] using two stages of learners, known as *base learners* and *meta learners* respectively. In the first step, multiple base learners were trained using the balanced training dataset. Each trained model was used to make predictions on the validation set. In the next step, the validation set was used to train the meta learner. In this case, however, the feature vector was augmented by concatenating the predictions earlier made by the base learners. These predictions thus form yet another feature descriptor for the meta learner, which we refer to as *base learner predictions (BLP)*.

We experimented with ten machine learning algorithms: Naive Bayes (NB), K-nearest Neighbor (KNN), Logistic Regression (LR), Partial Least Square Regression (PLS), Support Vector Machine (SVM), Multi-layer Perceptron (MLP), Decision Tree (DT), Random Forest (RF), Extra Tree (ET) and Extreme Gradient Boosting (XGB). We analysed the mutual information (MI) [53] between the residue-wise prediction probabilities and the target variable for each of these learners. As SVM had the highest MI, it was added in both the base and meta layers.

Subsequently more predictors were added to the base layer in a phased approach as follows. At each phase, we tried to augment the current set of predictors with another predictor and calculated the MI between average prediction probabilities and labels. The predictor for which the average MI became the highest in the phase was added. It is possible that the MI may decrease in a phase from earlier phase. Nevertheless the updated set of predictors is used in the next phase and the process repeats. Finally the set of predictors that resulted in the highest MI across all the phases is chosen. Accordingly, four classifier, SVM, MLP, ET and PLS were selected as the base learners (see Results Section).

2.6. Imbalanced data handling

The datasets used in this study were significantly imbalanced. Therefore, it was necessary to balance the training dataset. We applied an ensemble random undersampling approach, inspired by [54] as follows. We prepared 10 subsampled datasets each of which contained all positive samples. As to the negative samples, four of these datasets contained 0.2% of the negative samples, randomly selected, three datasets contained 2% and the remaining three datasets contained randomly selected 20% of the negative samples. Using each of these 10 subsampled datasets, we trained separate SVM, MLP, ET and PLS models. The validation set was featurized using selected feature descriptors, then augmented with the output of the base layer models, which we have referred to as the BLP descriptor. It was then used to train the SVM model in the meta layer. Data balancing in the meta layer was done using random undersampling.

2.7. Hyperparameter tuning

We tuned the hyperparameters of the base and meta layer classifiers to optimize prediction performance. We only tuned the crucial hyperparameters. The rest were set at their default values. We used *GridSearchCV* from the *scikit-learn* package in Python to grid search the hyperparameter space. We selected the values of each selected parameters of each classifiers that had balanced performance across all performance metrics. After tuning the base classfiers, we applied the same process to the meta layer classifer on validation dataset.

3. Results

3.1. Embedding from T5 model outperforms other embeddings

The 10-fold CV results, averaged across all 10 predictors trained on protein embeddings from the six different pre-trained language models are shown in Table 2. ProtT5-XL-U50 embedding wins with respect to all the performance metrics. The other T5 model, trained on the BFD corpora, came out second best. Therefore we selected the ProtT5-XL-U50 embedding for our model.

3.2. Embedding from language model wins over traditional feature descriptors

We wanted to examine how each feature descriptor influences the prediction of protein-carbohydrate binding sites. For each feature descriptor, we trained ten ML algorithms as stated before - NB, KNN, LR, PLS, SVM, MLP, DT, RF, ET and XGB. The averages across these classifiers were used to compare the predictive ability of the descriptors. As observed from Table 3, the embedding from ProtT5-XL-U50 exhibited the highest average scores across all performance metrics, indicating its effectiveness in predicting protein-carbohydrate binding sites. DPC is the best descriptor among the traditional descriptors, followed by PSSM.

Table 2: 10-fold CV performance of embeddings from various pre-trained protein language models. Results are averaged across models trained using various learning algorithms on the benchmark dataset balanced with random undersampling.

Language Model	**SN**	**SP**	**BACC**	**Acc**	**Prec**	**F1**	**MCC**	**AUC**	**auPR**
ProtBert	0.697	0.719	0.708	0.708	0.715	0.704	0.418	0.775	0.766
ProtBert-BFD	0.712	0.718	0.715	0.715	0.717	0.714	0.431	0.780	0.773
ProtAlbert	0.726	0.726	0.726	0.726	0.728	0.726	0.454	0.792	0.784
ProtXLNet	0.701	0.579	0.640	0.640	0.630	0.661	0.284	0.682	0.650
ProtT5-XL-BFD	0.730	0.772	0.751	0.751	0.765	0.745	0.505	0.819	0.811
ProtT5-XL-U50	**0.738**	**0.787**	**0.762**	**0.762**	**0.779**	**0.756**	**0.528**	**0.830**	**0.825**

Table 3: 10-fold CV performance of various feature descriptors. Results are averaged across models trained using various learning algorithms on the benchmark dataset balanced with random undersampling.

Feature Set	**SN**	**SP**	**BACC**	**Acc**	**Prec**	**F1**	**MCC**	**AUC**	**auPR**
PSSM	0.612	0.679	0.645	0.645	0.657	0.631	0.293	0.697	0.690
MonoGram	0.617	0.524	0.571	0.571	0.568	0.588	0.143	0.596	0.576
DPC	0.633	0.676	0.654	0.654	0.660	0.644	0.310	0.713	0.704
ASA	0.581	0.494	0.537	0.537	0.534	0.554	0.075	0.545	0.531
HSE	0.581	0.546	0.564	0.564	0.560	0.569	0.129	0.595	0.584
Torsion_angles	0.591	0.478	0.535	0.535	0.531	0.556	0.072	0.548	0.541
Physicochemical	0.445	0.576	0.511	0.511	0.491	0.453	0.021	0.510	0.515
ProtT5-XL-U50	**0.738**	**0.787**	**0.762**	**0.762**	**0.779**	**0.756**	**0.528**	**0.830**	**0.825**

3.3. Protein embedding along with PSSM constitutes the optimal feature descriptor

We had seven traditional feature descriptors and one embedding feature descriptor (ProtT5-XL-U50) at our disposal. As described earlier, to identify the optimal subset of features from this set we employed incremental feature selection (Algorithm 1). Table 4 shows the results of adding different feature groups. Based on the F1-score, we find the combination of ProtT5-XL-U50 embedding and PSSM to be the optimal feature descriptor.

3.4. SVM, MLP, ET and PLS constitute the base layer of StackCBEmbed

Table 5 reports the highest MI obtained in each phase and the corresponding learner combination. When multiple combinations in a phase resulted in very close MI values, all combinations were considered for exploration in the next phase. For example, in phase 3, {SVM, MLP, RF} and {SVM, MLT, ET} produced very close MI values. So we expanded the search in phase 4 based on both these combinations. Based on the overall results, SVM, MLP, ET and PLS were selected as the base learners. The results from the full search for base layer learners is available in Supplementary file.

3.5. SVM is chosen as the Meta layer classifier

As SVM had the highest MI score (Table 5), it was also chosen as the meta layer classifier. Since the meta layer will be trained on the validation set, we further evaluated all the classifiers using 10-fold CV and the results are shown in Table 6. SVM has the most balanced performance. Therefore, our choice of SVM for the meta layer has been well-justified.

Table 4: 10-fold CV performance of XGB predictor trained on the benchmark dataset (balanced with random undersampling) using various feature subsets as per Algorithm 1. An extra horizontal line separates each increment phase. ✓mark indicates that the row was selected for the next increment phase (based on F1-score). The best F1-score across all subsets is bold-faced.

Feature combination	**SN**	**SP**	**BACC**	**Acc**	**Prec**	**F1**	**MCC**	**AUC**	**auPR**	
PSSM	0.641	0.677	0.659	0.659	0.666	0.653	0.318	0.726	0.726	
MonoGram	0.576	0.550	0.563	0.563	0.562	0.568	0.127	0.603	0.613	
DPC	0.670	0.685	0.678	0.678	0.681	0.675	0.356	0.758	0.761	
ASA	0.561	0.468	0.515	0.515	0.515	0.536	0.030	0.521	0.516	
HSE	0.605	0.549	0.577	0.577	0.573	0.588	0.154	0.608	0.595	
Torsion_angles	0.564	0.516	0.540	0.540	0.540	0.551	0.080	0.568	0.554	
Physicochemical	0.492	0.535	0.514	0.514	0.514	0.501	0.027	0.516	0.521	
ProtT5-XL-U50	0.754	0.810	0.782	0.782	0.800	0.776	0.566	0.864	0.875	✓
ProtT5-XL-U50, PSSM	0.776	0.814	0.795	0.795	0.806	**0.791**	0.591	0.869	0.879	✓
ProtT5-XL-U50, MonoGram	0.759	0.804	0.782	0.782	0.795	0.776	0.564	0.868	0.880	
ProtT5-XL-U50, DPC	0.772	0.801	0.786	0.787	0.795	0.783	0.574	0.869	0.878	
ProtT5-XL-U50, ASA	0.758	0.815	0.787	0.787	0.806	0.781	0.576	0.869	0.877	
ProtT5-XL-U50, HSE	0.773	0.801	0.787	0.787	0.795	0.784	0.575	0.865	0.875	
ProtT5-XL-U50, Torsion_angles	0.766	0.816	0.791	0.791	0.806	0.785	0.583	0.865	0.874	
ProtT5-XL-U50, Physicochemical	0.766	0.812	0.789	0.789	0.804	0.784	0.579	0.866	0.877	
ProtT5-XL-U50, PSSM, MonoGram	0.769	0.815	0.792	0.792	0.808	0.787	0.586	0.867	0.879	
ProtT5-XL-U50, PSSM, DPC	0.767	0.817	0.792	0.792	0.810	0.787	0.587	0.870	0.882	
ProtT5-XL-U50, PSSM, ASA	0.764	0.807	0.785	0.786	0.799	0.781	0.572	0.869	0.879	
ProtT5-XL-U50, PSSM, HSE	0.778	0.804	0.791	0.791	0.799	0.788	0.582	0.864	0.876	
ProtT5-XL-U50, PSSM, Torsion_angles	0.771	0.816	0.793	0.793	0.808	0.788	0.588	0.870	0.878	✓
ProtT5-XL-U50, PSSM, Physicochemical	0.763	0.819	0.791	0.791	0.808	0.785	0.583	0.862	0.875	
ProtT5-XL-U50, PSSM, Torsion_angles, MonoGram	0.770	0.814	0.792	0.792	0.807	0.787	0.585	0.868	0.879	✓
ProtT5-XL-U50, PSSM, Torsion_angles, DPC	0.762	0.803	0.782	0.783	0.795	0.778	0.566	0.870	0.879	
ProtT5-XL-U50, PSSM, Torsion_angles, ASA	0.768	0.805	0.786	0.787	0.798	0.782	0.574	0.864	0.874	
ProtT5-XL-U50, PSSM, Torsion_angles, HSE	0.761	0.802	0.782	0.782	0.794	0.777	0.564	0.861	0.878	
ProtT5-XL-U50, PSSM, Torsion_angles, Physicochemical	0.765	0.818	0.792	0.792	0.809	0.786	0.585	0.863	0.871	
ProtT5-XL-U50, PSSM, Torsion_angles, MonoGram, DPC	0.771	0.818	0.795	0.795	0.810	0.789	0.591	0.871	0.878	✓
ProtT5-XL-U50, PSSM, Torsion_angles, MonoGram, ASA	0.765	0.807	0.786	0.786	0.798	0.781	0.573	0.864	0.874	
ProtT5-XL-U50, PSSM, Torsion_angles, MonoGram, HSE	0.770	0.804	0.787	0.787	0.797	0.783	0.575	0.871	0.881	
ProtT5-XL-U50, PSSM, Torsion_angles, MonoGram, Physicochemical	0.769	0.806	0.787	0.788	0.799	0.783	0.576	0.862	0.874	
ProtT5-XL-U50, PSSM, Torsion_angles, MonoGram, DPC, ASA	0.769	0.822	0.796	0.796	0.813	0.790	0.593	0.872	0.877	✓
ProtT5-XL-U50, PSSM, Torsion_angles, MonoGram, DPC, HSE	0.773	0.811	0.792	0.792	0.803	0.787	0.585	0.870	0.871	
ProtT5-XL-U50, PSSM, Torsion_angles, MonoGram, DPC, Physicochemical	0.773	0.812	0.792	0.792	0.805	0.788	0.586	0.868	0.876	
ProtT5-XL-U50, PSSM, Torsion_angles, MonoGram, DPC, ASA, HSE	0.755	0.814	0.784	0.785	0.803	0.778	0.571	0.871	0.877	
ProtT5-XL-U50, PSSM, Torsion_angles, MonoGram, DPC, ASA, Physicochemical	0.769	0.817	0.793	0.793	0.809	0.788	0.588	0.870	0.873	✓
ProtT5-XL-U50, PSSM, Torsion_angles, MonoGram, DPC, ASA, Physicochemical, HSE	0.765	0.819	0.792	0.792	0.808	0.786	0.585	0.874	0.879	

Table 5: Mutual information (MI) between the predictions and labels for the different subsets of predictors. For each subset, the probability of binding site is averaged across the predictors. All the predictors were trained on the training dataset balanced with random undersampling.

Phase	Best Learner Combination	MI
1	SVM	0.0430
2	SVM, MLP	0.0495
3a	SVM, MLP, RF	0.0492
4a	SVM, MLP, RF, KNN	0.0497
5a	SVM, MLP, RF, KNN, XGB	0.0511
6a	SVM, MLP, RF, KNN, XGB, ET	0.0516
7a	SVM, MLP, RF, KNN, XGB, ET, NB	0.0487
8ab	SVM, MLP, RF, KNN, XGB, ET, NB, LR	0.0445
8ac	SVM, MLP, RF, KNN, XGB, ET, NB, PLS	0.0445
9a	SVM, MLP, RF, KNN, XGB, ET, NB, LR, PLS	0.0441
10a	SVM, MLP, RF, KNN, XGB, ET, NB, PLS, LR, DT	0.0434
3b	SVM, MLP, ET	0.0491
4b	**SVM, MLP, ET, PLS**	**0.0521**
5b	SVM, MLP, ET, PLS, RF	0.0513
6b	SVM, MLP, ET, PLS, RF, KNN	0.0488
7b	SVM, MLP, ET, PLS, RF, KNN, LR	0.0495
8b	SVM, MLP, ET, PLS, RF, KNN, LR, NB	0.0498
9b	SVM, MLP, ET, PLS, RF, KNN, LR, NB, DT	0.0452

Figure 1 captures the overall training process of StackCBEmbed. The workflow for predicting the binding sites in a query protein is demonstrated in Figure 2.

Table 6: 10-fold CV performance of models trained on the validation set using embedding and PSSM features. The validation set was balanced using random undersampling.

Classifier	SN	SP	BACC	Acc	Prec	F1	MCC	AUC	auPR
RF	0.775	**0.829**	0.802	0.802	0.822	0.795	0.609	0.872	0.883
ET	0.786	0.822	0.804	0.804	0.819	0.799	0.611	**0.881**	**0.889**
DT	0.691	0.695	0.693	0.693	0.695	0.692	0.387	0.693	0.636
MLP	**0.840**	0.692	0.766	0.765	0.734	0.782	0.540	0.837	0.845
LR	0.742	0.732	0.737	0.736	0.738	0.737	0.476	0.817	0.824
SVM	0.815	0.801	**0.808**	**0.807**	0.807	**0.809**	0.618	0.877	0.884
NB	0.767	0.811	0.789	0.789	0.803	0.782	0.582	0.836	0.796
KNN	0.720	0.790	0.755	0.755	0.776	0.744	0.514	0.807	0.790
XGB	0.782	0.**829**	0.806	0.805	**0.826**	0.797	**0.619**	0.873	0.875
PLS	0.793	0.794	0.793	0.793	0.800	0.793	0.590	0.867	0.876

3.6. Hyperparameter tuning

The hyperparameters for the base layer classifiers (SVM, MLP, ET, PLS) and the meta layer classifier (SVM) were tuned using *GridSearchCV*. The parameters we tuned, alongside their final values, are given in Table 7. (See Supplementary file for the full grid search results).

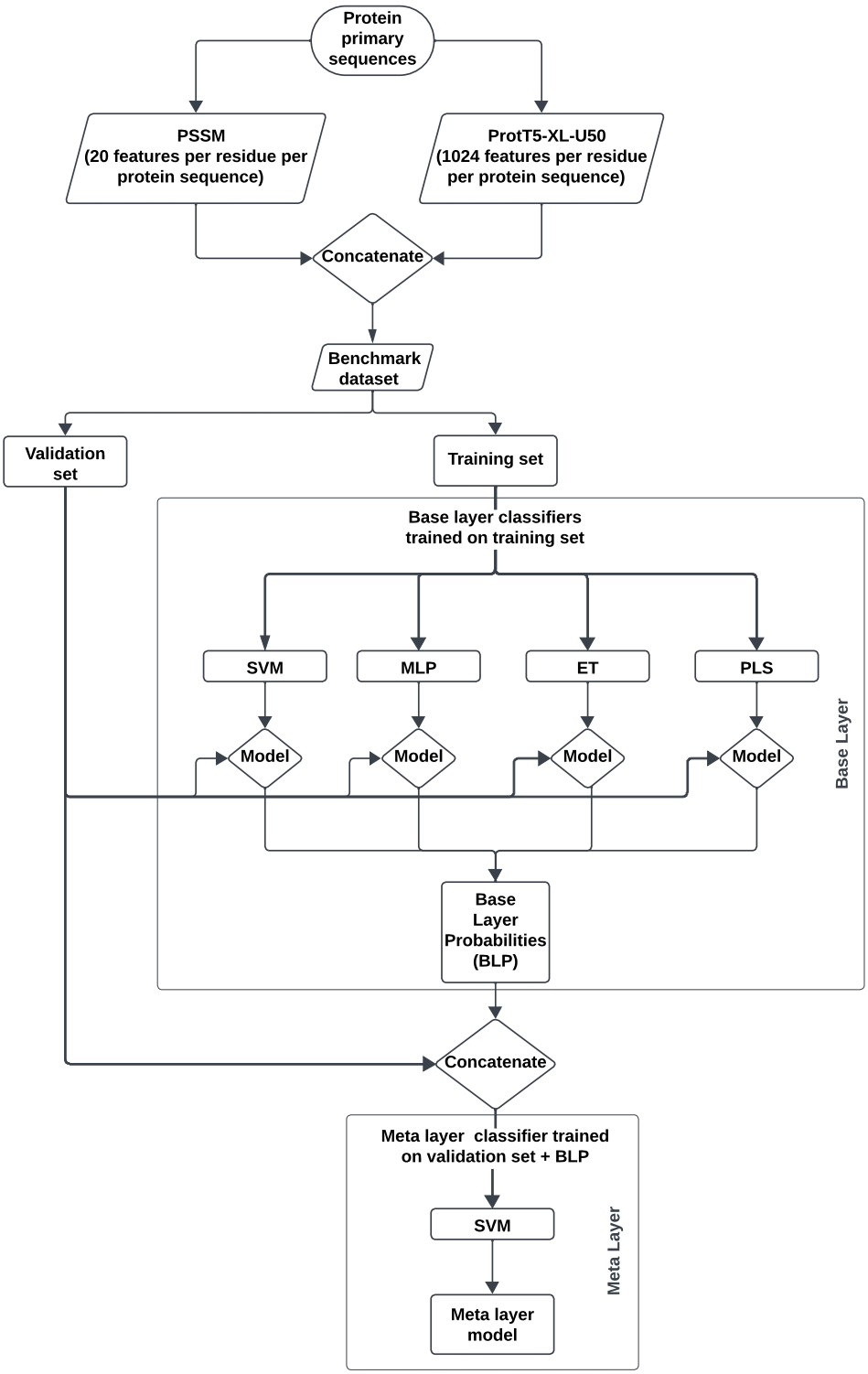


Figure 1: The framework and training process of StackCBEmbed.

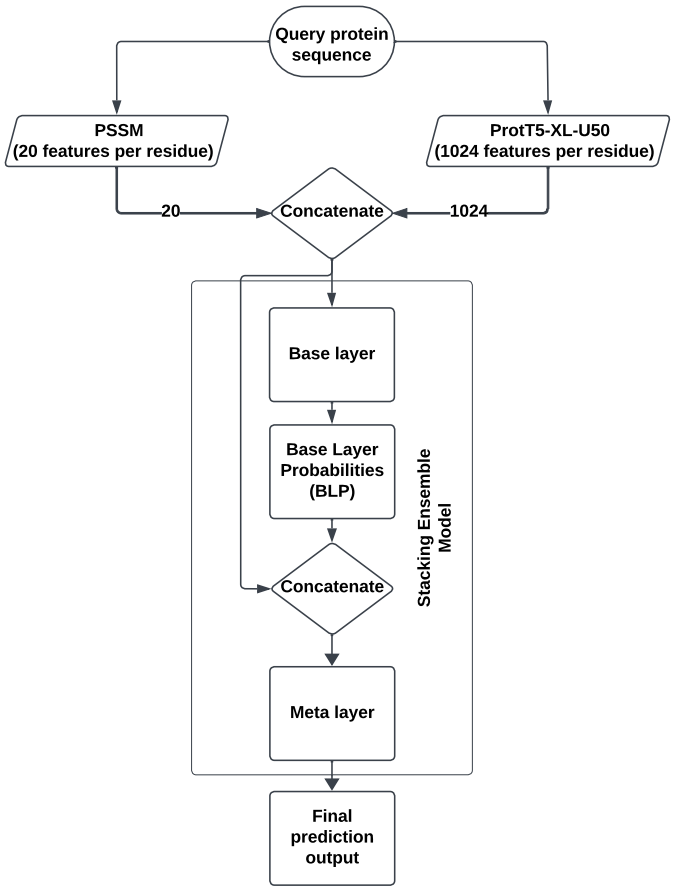


Figure 2: StackCBEmbed workflow for protein-carbohydrate binding-site prediction for a novel/query protein.

3.7. Stacking ensemble significantly improves prediction performance

The best 5 feature combinations from Table 4, with respect to F1-score, are highlighted in Table 8. In addition, FC-6, is the augmentation of FC-1 with the predictions obtained from the base layer predictors. In our stacking ensemble model, FC-6 is what the SVM learner in the meta layer learns on.

To examine the efficacy of stacking, we checked the performance of six SVM models trained using the validation set. Each model was trained using one of the six feature combinations mentioned in Table 8. The 10-fold CV performance comparison is shown in Figure 3. The FC-6 feature combination, which represents stacking, clearly outperforms the other combinations with respect to most of the performance metrics. Additionally, we used the t-distributed stochastic neighbor embedding (t-SNE) [55] to compare the feature space distribution of the feature combinations of Table 8 on the balanced validation dataset. As can be observed from Figure 4, the feature space of FC-6 produces a better separation between red and blue spots. This explains the superior predictive ability of stacking ensemble compared to the other baseline models.

3.8. StackCBEmbed outperforms state-of-the-art methods

We have used two independent test sets, TS49 and TS88, to compare the performance of StackCBEmbed with two state-of-the-art (SOTA) methods, SPRINT-CBH and StackCBPred. The

Table 7: List of hyperparameters with their tuned values

Layer	Classifier	Parameters	Values
Base layer	SVM	**C**	10
		degree	3
		gamma	auto
		kernel	rbf
	MLP	**activation**	relu
		alpha	0.05
		hidden_layer_sizes	(50, 50, 50)
		learning_rate	constant
		solver	adam
	ET	**bootstrap**	False
		criterion	gini
		max_depth	20
		max_features	sqrt
		n_estimators	200
	PLS	**n_components**	2
		scale	True
		max_itr	500
		tol	1e-06
Meta layer	SVM	**C**	1
		degree	3
		gamma	scale
		kernel	rbf

Table 8: Top 5 feature combinations from Table 4 in terms of F1-score. These combinations have been named from FC-1 to 5 for ease of reference in subsequent experiments. FC-6 is the augmentation of FC-1 with the base learner prediction values.

Name	Feature Combination
FC-1	ProtT5-XL-U50, PSSM
FC-2	ProtT5-XL-U50, PSSM, TA, MG, DPC, ASA
FC-3	ProtT5-XL-U50, PSSM, TA, MG, DPC
FC-4	ProtT5-XL-U50, PSSM, TA
FC-5	ProtT5-XL-U50, PSSM, TA, MG, DPC, PHYSICO
FC-6	ProtT5-XL-U50, PSSM, BLP

SPRINT-CBH and StackCBPred servers were inaccessible, preventing us from reporting some of the metrics. Nevertheless the trained model of StackCBPred was accessible in its git repository, enabling us to replicate the results. We have presented both the published values as well as the reproduced values of performance metrics in Tables 9 and 10.

While StackCBEmbed uses only two feature descriptors, PSSM and embedding, PSSM is expensive to compute. Therefore, we created two other models, one using PSSM only, and the other one only using the embedding. We wanted to check how well the model performs in case PSSM for a query protein cannot be provided. From the results in Tables 9 and 10, we can see that when only PSSM is used, the model produces poor results in both the datasets. However, when it uses embedding only, the results are quite good. In both datasets, it beats SOTA methods in terms of sensitivity, balanced accuracy, F1-score, AUC and auPR. In TS49 it also beat SOTA methods with respect to MCC. When both the feature descriptors are combined, the results are further improved.

In TS49, StackCBEmbed with only the embedding features improved the sensitivity by $\sim 89.2\%$

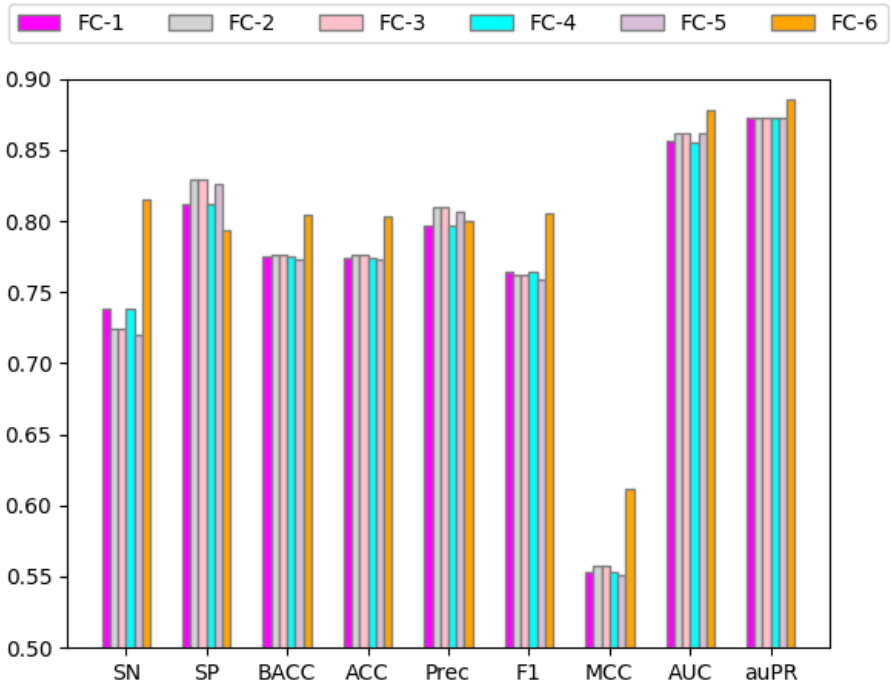


Figure 3: 10-fold CV performance of SVM models trained on the validation set for various feature combinations. The validation set was balanced using random undersampling prior to training.

Table 9: Comparison of StackCBEmbed with SOTA methods on the TS49 testset.

Predictor	**SN**	**SP**	**BACC**	**ACC**	**F1**	**MCC**	**AUC**	**auPR**
StackCBPred (as per [20])	0.553	0.795	0.674	0.786	-	0.159	-	-
StackCBPred (reproduced)	0.522	0.803	0.662	0.792	0.157	0.151	0.725	0.106
SPRINT-CBH	0.389	**0.925**	0.657	**0.906**	-	0.195	0.755	-
StackCBEmbed (PSSM)	0.553	0.742	0.647	0.735	0.134	0.125	0.706	0.085
StackCBEmbed (Embedding)	**0.736**	0.817	**0.777**	0.814	0.227	0.260	**0.834**	**0.237**
StackCBEmbed	0.730	0.821	0.776	0.818	**0.229**	**0.261**	0.832	0.233

Table 10: Comparison of StackCBEmbed with SOTA methods on the TS88 testset.

Predictor	**SN**	**SP**	**BACC**	**ACC**	**F1**	**MCC**	**AUC**	**auPR**
StackCBPred (as per [20])	0.565	0.797	0.681	-	-	0.139	-	-
StackCBPred (reproduced)	0.499	0.750	0.624	0.744	0.090	0.090	0.671	0.055
SPRINT-CBH	0.130	**0.997**	0.564	-	-	**0.257**	-	-
StackCBEmbed (PSSM)	0.494	0.740	0.617	0.733	0.086	0.083	0.668	0.047
StackCBEmbed (Embedding)	0.657	0.818	0.738	**0.814**	0.152	0.189	0.784	0.136
StackCBEmbed	**0.666**	0.818	**0.742**	**0.814**	**0.154**	0.192	**0.788**	**0.141**

and $\sim$ 33.1%, compared to SPRINT-CBH and StackCBPred respectively. On the other hand, with both PSSM and embedding features it improved on the BACC compared to the existing SPRINT-CBH ($\sim$ 18.3%) and StackCBPred ($\sim$ 15.3%). On the other hand, in the TS88 dataset, StackCBEmbed outperforms SPRINT-CBH and StackCBPred methods with an improved sensitivity of

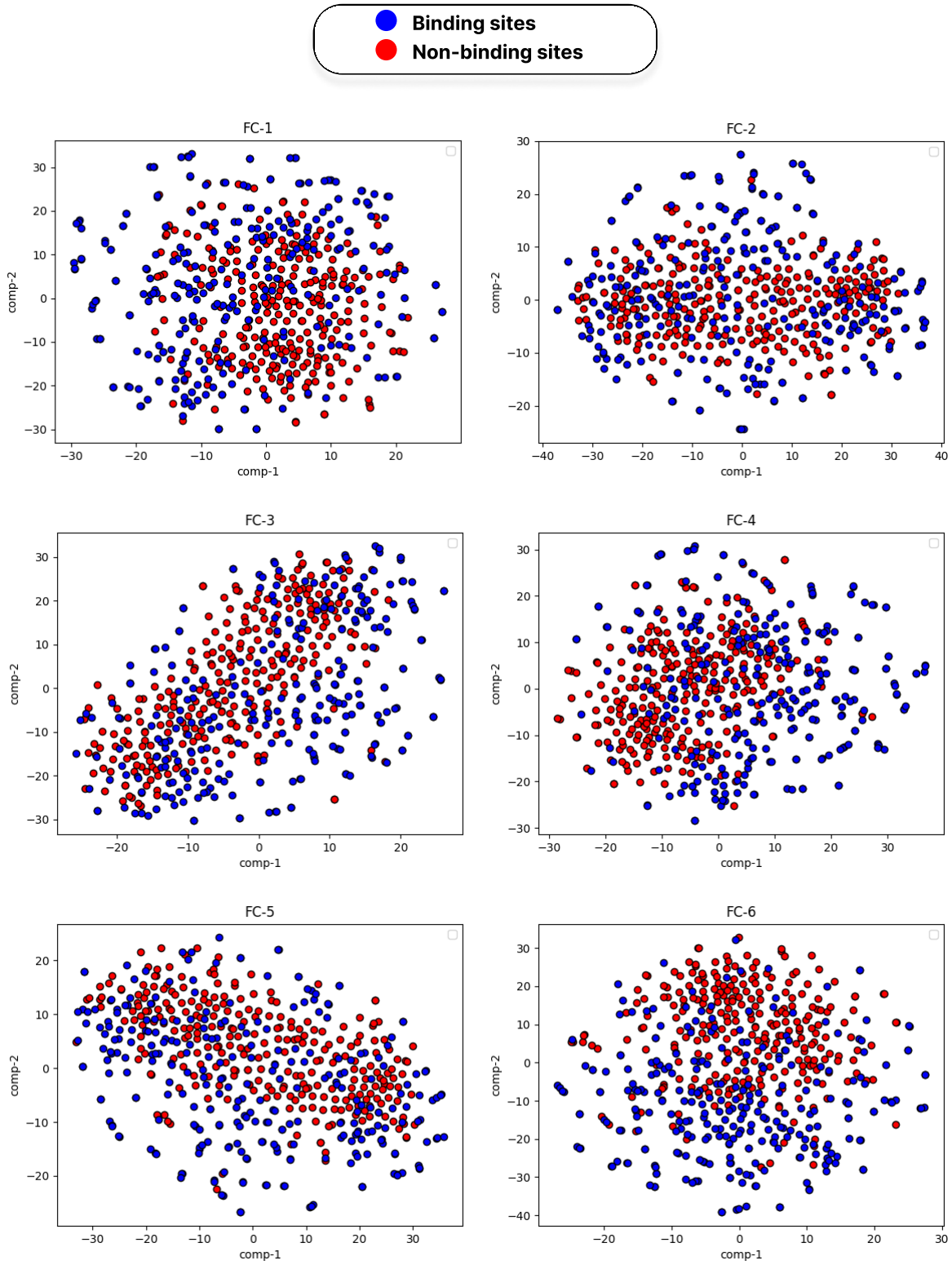
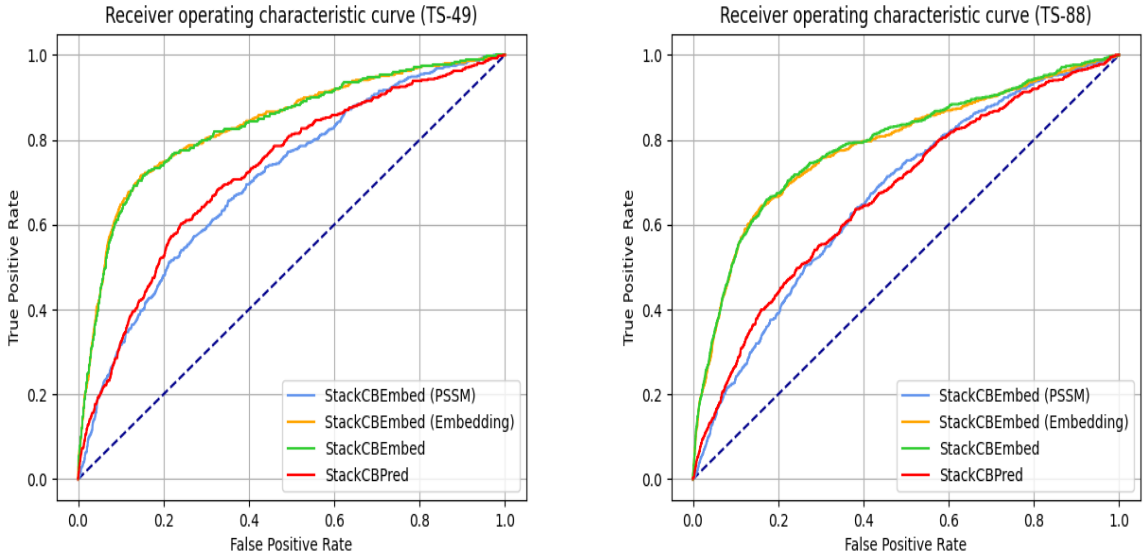


Figure 4: Six t-SNE plots on the validation dataset reflect the distribution of binding sites (blue dots) and non-binding sites (red dots) for the six feature combinations mentioned in Table 8. The feature space corresponding FC-6 produces a much clearer separation between the binding and non-binding sites.

Figure 5: Comparison of Receiver Operating Characteristic (ROC) curves on TS49 and TS88 test sets.



$\sim$ 412.3% and $\sim$ 17.9% respectively. We also observe a significant improvement in BACC compared to SPRINT-CBH ($\sim$ 31.6%) and StackCBPred ($\sim$ 9%). Although it achieved a good improvement in MCC value ($\sim$ 38.1%) compared to StackCBPred method, it could not exceed the MCC value of SPRINT-CBH.

The Receiver Characteristic curve (ROC) represents the true positive rate (TPR) and false positive rate (FPR) values at several classification thresholds. Figure 5 shows the ROC curves (in independent test sets) for different flavors of StackCBEmbed. It also plots the same for StackCBPred. From the curves in both datasets, it is clear that PSSM alone cannot give enough predictive capability to StackCBEmbed. However, when embedding feature is added, its performance improves drastically, beating that of StackCBPred. When embedding alone is used, the performance is very close to the regular StackCBEmbed.

3.9. The difference between StackCBEmbed and StackCBPred is statistically significant

There is significant methodological differences between StackCBPred and StackCBEmbed. Firstly, our approach in StackCBEmbed involved integrating traditional sequence-based features with recently introduced protein language model based embedding features. StackCBPred, however, exclusively relied on traditional sequence-based features. The inclusion of embedding features have enhanced the representation and comprehension of the data within our model. Secondly, in StackCBEmbed, we implemented an ensemble sampling architecture to address the challenge posed by imbalanced datasets. In contrast, StackCBPred simply used random undersampling, a technique with the potential to discard valuable information. Lastly, our choice of base classifiers in StackCBEmbed was guided by mutual-information gain scores, whereas StackCBPred included classifiers with differing underlying learning principles. Our objective approach in classifier selection in the base layer could have influenced the discriminative power of our model, potentially resulting in improved predictive outcomes.

We conducted McNemar's test on both TS49 and TS88 to check whether the differences between the two methods are statistically significant or not. The null hypothesis assumed no distinction between StackCBEmbed and StackCBPred methods. A significance level (alpha) of 0.05 was set.

Table 11: Contingency table for McNemar's test, comparing the predicted outcomes of StackCBEmbed and StackCBPred across the binding sites on TS49 test set.

		StackCBPred	
		Yes	No
StackCBEmbed	Yes	225	146
	No	40	97

Table 12: Contingency table for McNemar's test, comparing the predicted outcomes of StackCBEmbed and StackCBPred across all sites on TS49 test set.

		StackCBPred	
		Yes	No
StackCBEmbed	Yes	1274	1462
	No	1600	9402

Table 13: Contingency table for McNemar's test, comparing the predicted outcomes of StackCBEmbed and StackCBPred across the binding sites on TS88 test set.

		StackCBPred	
		Yes	No
StackCBEmbed	Yes	271	187
	No	72	158

Table 14: Contingency table for McNemar's test, comparing the predicted outcomes of StackCBEmbed and StackCBPred across all sites on TS88 test set.

		StackCBPred	
		Yes	No
StackCBEmbed	Yes	2601	2674
	No	4348	17518

Four contingency tables, Table 11, Table 12, Table 13, and Table 14, were created based on the paired data, each containing four cells representing different outcomes. The McNemar test statistic was computed using the formula $\chi^2 = (b - c)^2/(b + c)$ which follows a chi-square distribution with 1 degree of freedom. Finally, the obtained p-value from McNemar's test was used to accept or reject the null hypothesis. The p-values obtained for Tables 11, 12, 13 and 14 were 8×10^{-15}, 1.2635333×10^{-2}, 8.95×10^{-13}, and 0, respectively. These point to a statistically significant difference between StackCBEmbed and StackCBPred.

4. Discussion and Conclusion

In this study, we have proposed a novel prediction model for protein-carbohydrate binding sites. Named StackCBEmbed, the predictor is a stacking ensemble model trained using 98 protein sequences ($\leq$ 30% pairwise sequence identity), comprising 1014 binding and 25705 non-binding residues. 70% of this data (training set) was used to train the base layer, while the remaining 30% (validation set) was used in training the meta layer. From a variety of traditional sequence-based features, and embedding from recently trained protein language models, optimal feature vector were prepared using incremental feature selection. In the final model a combination of PSSM and embedding features were used. As the training data is heavily imbalanced, for improved learning, 10 different subsamples was prepared, each with a different ratio of positive and negative samples. The base layer of the model used SVM, MLP, ET and PLS learners. As the meta layer, an SVM classifier was trained on the validation set, augmented with the prediction probabilities from the base layer models, and balanced using random undersampling. Two independent test datasets were used for performance assessment and comparison with SOTA methods.

Results in both the independent test sets indicate that StackCBEmbed outperforms SOTA methods, such as SPRINT-CBH and StackCBPred, even when only the embedding features are used. The results further improve if PSSM features are added. This is an important observation because PSSM cannot be computed for proteins with not enough homologs in the database. Also, for proteins with enough homologs, the process to compute PSSM is time consuming. Therefore, a researcher can choose to avoid using PSSM and still get good prediction results using StackCBEmbed.

We also looked at another SOTA method stemming out of the work by Malik et al. [6]. This publication is quite old and uses a dataset of 40 protein-carbohydrate complexes only. Neither a prediction server nor a standalone downloadable version was available for this method. Therefore we opted to exclude this approach from our benchmarking. More recent methods, such as, LMNGlypred [56] and PUStackNGly [57] were also reviewed. But these models are primarily designed for predicting N-linked glycosylation sites. Our work, on the other hand, focuses on identifying non-specific carbohydrate binding sites, rendering these methods out of scope for comparative analysis.

Prediction models in different problems in bioinformatics are dominated by deep learning methods in the modern era. Accordingly, we started our exploration with several deep learning architectures alongside the traditional ML algorithms. Specifically, we investigated LSTM [58], ResNet [59] which did not produce promising results. This might have been due to the significant imbalance in the dataset. As we achieved early success in the traditional approaches, using embeddings from pre-trained models, we did not pursue other DL methods any further. The various recent DL architectures, suitable for sequence data, should be further investigated in future research.

While StackCBEmbed outperformed SOTA methods in most metrics in both the independent test sets, it did not surpass the MCC value of SPRINT-CBH in TS88 dataset. However, a recent research [60] notes that MCC is not a suitable metric for imbalanced datasets due to its performance inconsistency, skewed behavior in ROC space, sensitivity to threshold values, and the clustering of values around the middle of the range. This emphasizes the necessity for alternative metrics and careful interpretation. Nevertheless analyzing F1 and auPR scores of StackCBEmbed reveals room for improvement, particularly in precision values, despite achieving promising sensitivity compared to state of the art. These findings highlight the importance of further research and development in this domain.

StackCBEmbed achieved a balanced accuracy and sensitivity of 77.6% and 73%, respectively in the TS49 test set. In case of TS88, these values were 74.2% and 66.6%, respectively. Thus StackCBEmbed brings in significant improvements in protein-carbohydrate binding site prediction. The model is freely available as an easy to use python script at `https://github.com/nafiislam/StackCBEmbed`. We hope that research related to protein-carbohydrate binding sites will immensely benefit from the efficacy of this novel, easy-to-use predictor.

References

[1] Clara Shionyu-Mitsuyama, Tsuyoshi Shirai, Hirokazu Ishida, and Takashi Yamane. An empirical approach for structure-based prediction of carbohydrate-binding sites on proteins. *Protein Engineering*, 16(7):467–478, 2003.

[2] Maria del Carmen Fernandez-Alonso, Dolores Díaz, Manuel Alvaro Berbis, Filipa Marcelo, Jesus Jimenez-Barbero, et al. Protein-carbohydrate interactions studied by nmr: from molecular recognition to drug design. *Current Protein and Peptide Science*, 13(8):816–830, 2012.

[3] Injae Shin, Sungjin Park, and Myung-ryul Lee. Carbohydrate microarrays: an advanced technology for functional studies of glycans. *Chemistry–A European Journal*, 11(10):2894–2901, 2005.

[4] Michaela Wimmerová, Stanislav Kozmon, Ivona Nečasová, Sushil Kumar Mishra, Jan Komárek, and Jaroslav Koča. Stacking interactions between carbohydrate and protein quantified by combination of theoretical and experimental methods. 2012.

[5] Ben Rathje, Caelen Begg, Liv Helland, and Pari Kyars. A review of common shoulder injuries: clavicular fractures and anterior dislocations. *MacEwan University Student eJournal*, 4(1), 2020.

[6] Adeel Malik and Shandar Ahmad. Sequence and structural features of carbohydrate binding in proteins and assessment of predictability using a neural network. *BMC Structural Biology*, 7:1–14, 2007.

[7] Alan Brown and Matthew K Higgins. Carbohydrate binding molecules in malaria pathology. *Current opinion in structural biology*, 20(5):560–566, 2010.

[8] KO François and Jan Balzarini. Potential of carbohydrate-binding agents as therapeutics against enveloped viruses. *Medicinal research reviews*, 32(2):349–387, 2012.

[9] Avraham Raz and Susumu Nakahara. Biological modulation by lectins and their ligands in tumor progression and metastasis. *Anti-Cancer Agents in Medicinal Chemistry (Formerly Current Medicinal Chemistry-Anti-Cancer Agents)*, 8(1):22–36, 2008.

[10] Ghazaleh Taherzadeh, Yaoqi Zhou, Alan Wee-Chung Liew, and Yuedong Yang. Sequence-based prediction of protein–carbohydrate binding sites using support vector machines. *Journal of chemical information and modeling*, 56(10):2115–2122, 2016.

[11] Chiara Taroni, Susan Jones, and Janet M Thornton. Analysis and prediction of carbohydrate binding sites. *Protein Engineering*, 13(2):89–98, 2000.

[12] MS Sujatha and Petety V Balaji. Identification of common structural features of binding sites in galactose-specific proteins. *Proteins: Structure, Function, and Bioinformatics*, 55(1):44–65, 2004.

[13] Mahesh Kulharia, Stephen J Bridgett, Roger S Goody, and Richard M Jackson. Inca-sitefinder: a method for structure-based prediction of inositol and carbohydrate binding sites on proteins. *Journal of Molecular Graphics and Modelling*, 28(3):297–303, 2009.

[14] Houssam Nassif, Hassan Al-Ali, Sawsan Khuri, and Walid Keirouz. Prediction of protein-glucose binding sites using support vector machines. *Proteins: Structure, Function, and Bioinformatics*, 77(1):121–132, 2009.

[15] Keng-Chang Tsai, Jhih-Wei Jian, Ei-Wen Yang, Po-Chiang Hsu, Hung-Pin Peng, Ching-Tai Chen, Jun-Bo Chen, Jeng-Yih Chang, Wen-Lian Hsu, and An-Suei Yang. Prediction of carbohydrate binding sites on protein surfaces with 3-dimensional probability density distributions of interacting atoms. *PLoS One*, 7(7):e40846, 2012.

[16] M Michael Gromiha, K Veluraja, and Kazuhiko Fukui. Identification and analysis of binding site residues in proteincarbohydrate complexes using energy based approach. *Protein and Peptide Letters*, 21(8):799–807, 2014.

[17] NR Siva Shanmugam, J Fermin Angelo Selvin, K Veluraja, and M Michael Gromiha. Identification and analysis of key residues involved in folding and binding of protein-carbohydrate complexes. *Protein and peptide letters*, 25(4):379–389, 2018.

[18] Priyadarshini P Pai and Sukanta Mondal. Mowgli: prediction of protein–mannose interacting residues with ensemble classifiers using evolutionary information. *Journal of Biomolecular Structure and Dynamics*, 34(10):2069–2083, 2016.

[19] Sandhya Agarwal, Nitish Kumar Mishra, Harinder Singh, and Gajendra PS Raghava. Identification of mannose interacting residues using local composition. *PLoS One*, 6(9):e24039, 2011.

[20] Suraj Gattani, Avdesh Mishra, and Md Tamjidul Hoque. Stackcbpred: A stacking based prediction of protein-carbohydrate binding sites from sequence. *Carbohydrate research*, 486:107857, 2019.

[21] Tristan Bepler and Bonnie Berger. Learning the protein language: Evolution, structure, and function. *Cell systems*, 12(6):654–669, 2021.

[22] Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, et al. ProtTrans: Toward Understanding the Language of Life Through Self-Supervised Learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(10):7112–7127, 2021.

[23] Huiying Zhao, Yuedong Yang, Mark von Itzstein, and Yaoqi Zhou. Carbohydrate-binding protein identification by coupling structural similarity searching with binding affinity prediction. *Journal of computational chemistry*, 35(30):2177–2183, 2014.

[24] Adeel Malik, Ahmad Firoz, Vivekanand Jha, and Shandar Ahmad. Procarb: a database of known and modelled carbohydrate-binding protein structures with sequence-based prediction tools. *Advances in bioinformatics*, 2010, 2010.

[25] Mark Johnson, Irena Zaretskaya, Yan Raytselis, Yuri Merezhuk, Scott McGinnis, and Thomas L Madden. Ncbi blast: a better web interface. *Nucleic acids research*, 36(suppl_2):W5–W9, 2008.

[26] In-Kwon Yeo and Richard A Johnson. A new family of power transformations to improve normality or symmetry. *Biometrika*, 87(4):954–959, 2000.

[27] Sumaiya Iqbal and Md Tamjidul Hoque. Pbrpredict-suite: a suite of models to predict peptide-recognition domain residues from protein sequence. *Bioinformatics*, 34(19):3289–3299, 2018.

[28] Avdesh Mishra, Pujan Pokhrel, and Md Tamjidul Hoque. Stackdppred: a stacking based prediction of dna-binding protein from sequence. *Bioinformatics*, 35(3):433–441, 2019.

[29] Ashis Kumer Biswas, Nasimul Noman, and Abdur Rahman Sikder. Machine learning approach to predict protein phosphorylation sites by incorporating evolutionary information. *BMC bioinformatics*, 11:1–17, 2010.

[30] Md Nasrul Islam, Sumaiya Iqbal, Ataur R Katebi, and Md Tamjidul Hoque. A balanced secondary structure predictor. *Journal of theoretical biology*, 389:60–71, 2016.

[31] Sumaiya Iqbal, Avdesh Mishra, and Md Tamjidul Hoque. Improved prediction of accessible surface area results in efficient energy function application. *Journal of theoretical biology*, 380:380–391, 2015.

[32] Ruchi Verma, Grish C Varshney, and GPS Raghava. Prediction of mitochondrial proteins of malaria parasite using split amino acid composition and pssm profile. *Amino acids*, 39:101–110, 2010.

[33] Stephen F Altschul, Warren Gish, Webb Miller, Eugene W Myers, and David J Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.

[34] Harsh Saini, Gaurav Raicar, Alok Sharma, Sunil Lal, Abdollah Dehzangi, Rajeshkannan Ananthanarayanan, James Lyons, Neela Biswas, and Kuldip K Paliwal. Protein structural class prediction via k-separated bigrams using position specific scoring matrix. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 18(4):474–479, 2014.

[35] Alok Sharma, James Lyons, Abdollah Dehzangi, and Kuldip K Paliwal. A feature extraction technique using bi-gram probabilities of position specific scoring matrix for protein fold recognition. *Journal of theoretical biology*, 320:41–46, 2013.

[36] Taigang Liu, Xiaoqi Zheng, and Jun Wang. Prediction of protein structural class for low-similarity sequences using support vector machine and psi-blast profile. *Biochimie*, 92(10):1330–1334, 2010.

[37] Eshel Faraggi, Tuo Zhang, Yuedong Yang, Lukasz Kurgan, and Yaoqi Zhou. Spine x: improving protein secondary structure prediction by multistep learning coupled with prediction of solvent accessible surface area and backbone torsion angles. *Journal of computational chemistry*, 33(3):259–267, 2012.

[38] Rhys Heffernan, Abdollah Dehzangi, James Lyons, Kuldip Paliwal, Alok Sharma, Jihua Wang, Abdul Sattar, Yaoqi Zhou, and Yuedong Yang. Highly accurate sequence-based prediction of half-sphere exposures of amino acid residues in proteins. *Bioinformatics*, 32(6):843–849, 2016.

[39] Rhys Heffernan, Kuldip Paliwal, James Lyons, Jaswinder Singh, Yuedong Yang, and Yaoqi Zhou. Single-sequence-based prediction of protein secondary structures and solvent accessibility by deep whole-sequence learning. *Journal of computational chemistry*, 39(26):2210–2216, 2018.

[40] Rhys Heffernan, Yuedong Yang, Kuldip Paliwal, and Yaoqi Zhou. 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):2842–2849, 2017.

[41] Jiangning Song, Hao Tan, Kazuhiro Takemoto, and Tatsuya Akutsu. Hsepred: predict half-sphere exposure from protein sequences. *Bioinformatics*, 24(13):1489–1497, 2008.

[42] Jens Meiler, Michael Müller, Anita Zeidler, and Felix Schmäschke. Generation and evaluation of dimension-reduced amino acid parameter representations by artificial neural networks. *Molecular modeling annual*, 7(9):360–369, 2001.

[43] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[44] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[45] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

[46] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[47] Douglas G Altman and J Martin Bland. Diagnostic tests. 1: Sensitivity and specificity. *BMJ: British Medical Journal*, 308(6943):1552, 1994.

[48] David MW Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*, 2020.

[49] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[50] Jens Keilwagen, Ivo Grosse, and Jan Grau. Area under precision-recall curves for weighted and unweighted data. *PloS one*, 9(3):e92209, 2014.

[51] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[52] Ruopeng Xie, Jiahui Li, Jiawei Wang, Wei Dai, André Leier, Tatiana T Marquez-Lago, Tatsuya Akutsu, Trevor Lithgow, Jiangning Song, and Yanju Zhang. Deepvf: a deep learning-based hybrid framework for identifying virulence factors using the stacking strategy. *Briefings in bioinformatics*, 22(3):bbaa125, 2021.

[53] Jorge R Vergara and Pablo A Estévez. A review of feature selection methods based on mutual information. *Neural computing and applications*, 24:175–186, 2014.

[54] Chun-Qiu Xia, Xiaoyong Pan, and Hong-Bin Shen. Protein–ligand binding residue prediction enhancement through hybrid deep heterogeneous learning of sequence and structure data. *Bioinformatics*, 36(10):3018–3027, 2020.

[55] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[56] Subash C Pakhrin, Suresh Pokharel, Kiyoko F Aoki-Kinoshita, Moriah R Beck, Tarun K Dam, Doina Caragea, and Dukka B Kc. Lmnglypred: prediction of human n-linked glycosylation sites using embeddings from a pre-trained protein language model. *Glycobiology*, 33(5):411–422, 2023.

[57] Alhasan Alkuhlani, Walaa Gad, Mohamed Roushdy, and Abdel-Badeeh M Salem. Pustackngly: positive-unlabeled and stacking learning for n-linked glycosylation site prediction. *IEEE Access*, 10:12702–12713, 2022.

[58] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, 31(7):1235–1270, 2019.

[59] Fengxiang He, Tongliang Liu, and Dacheng Tao. Why resnet works? residuals generalize. *IEEE transactions on neural networks and learning systems*, 31(12):5349–5362, 2020.

[60] Qiuming Zhu. On the performance of matthews correlation coefficient (mcc) for imbalanced dataset. *Pattern Recognition Letters*, 136:71–80, 2020.