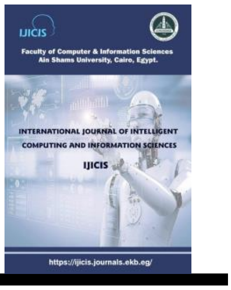
IJICIS, Vol.23, No.1, 41-52
DOI: 10.21608/ijicis.2023.160986.1218

International Journal of Intelligent Computing and Information Sciences

https://ijicis.journals.ekb.eg/


IJICIS
Faculty of Computer & Information Sciences Ain Shams University, Cairo, Egypt.
INTERNATIONAL JOURNAL OF INTELLIGENT COMPUTING AND INFORMATION SCIENCES
IJICIS
https://ijicis.journals.ekb.eg/

PREDICTION OF O-GLYCOSYLATION SITE USING PRE-TRAINED LANGUAGE MODEL AND MACHINE LEARNING

Alhasan Alkuhlani*

Computer Science
Department.,
Faculty of Computer and
Information Sciences, Ain
shams University,
Cairo, Egypt
Alhasan.alkuhlani@gmail.com

Walaa Gad

Information system
Department.,
Faculty of Computer and
Information Sciences,Ain
shams University,
Cairo, Egypt
walaagad@cis.asu.edu.eg

Mohamed Roushdy

Computer Science
Department.,
Faculty of Computers and
Information Technology,
Future University in Egypt,
Cairo, Egypt
mohamed.roushdy@fue.edu.eg

Abdel-badeeh M. Salem

Computer Science
Department.,
Faculty of Computer and
Information Sciences, Ain
shams University,
Cairo, Egypt
absalem@cis.asu.edu.eg

Received 2022-09-05; Revised 2023-12-24; Accepted 2023-02-27

Abstract: *O-glycosylation is a typical type of protein post-translational modifications (PTMs), which is linked to several diseases and has significant roles in many biological processes. Identification of O-glycosylation sites is important to know the mechanism of the O-glycosylation process. However, the identification of PTM sites by laboratory experimental tools is time and money-consuming. Thus, the utilization of computational and artificial intelligence is becoming essential to predict O-glycosylation sites. In this paper, we proposed a new model to improve O-glycosylation site prediction using a transformer-based protein language model and machine learning. The dataset was collected and prepared from a recent data source called OGP (O-glycoprotein repository). The TAPE (Tasks Assessing Protein Embeddings) protein language model was used to feature extraction from the peptide sequences using the embedding strategy. Then, feature selection was implemented using the linear support vector machine (SVM) to select informative features. The XGBoost ensemble-based machine learning method was utilized for classification and prediction. The proposed model achieved high-performance results with 0.7761 accuracy, 0.7391 sensitivity, 0.8130 specificity, 0.8295 AUC, and 0.5537 MCC when compared with the traditional machine learning methods. On an independent dataset, the proposed method performed better than the latest available methods for predicting O-glycosylation sites.*

Keywords: *protein language model, machine learning, XGBoost, Bioinformatics, O-glycosylation site prediction.*

1. Inroduction

1.1. Overview

***Corresponding Author**: Alhasan Alkuhlani

Computer Science Department, Faculty of Computer and Information Sciences, Ain shams University, Cairo, Egypt

Email address: Alhasan.alkuhlani@gmail.com

Post-translational modifications (PTMs) of proteins are the chemical alterations that take place after the protein is produced. Protein PTMs are fundamental for the structure, maturation, and functions of proteins. Thus, detecting and comprehending PTMs is crucial in cell biology research as well as disease prevention and treatment [1]. Glycosylation is a common type of PTMs in which a complex group of glycan is enzymatically linked to the amino acids of proteins. Recent studies reported that abnormal glycosylation causes various diseases like cancer, diabetes, and immunity diseases [2–4].

One of the major types of glycosylation is O-glycosylation, in which a glycan is joined to the hydroxyl group of a protein's serine (S) or threonine (T) amino acid. Prediction of glycosylation sites aids in understanding the biological process of glycosylation as well as helps in the treatment of diseases that are associated with it [2,5]. In order to improve O-glycosylation site identification as well as to reduce experimental effort and cost, computational intelligence and machine learning techniques have been considered for O-glycosylation site prediction. Bioinformatics applications are increasingly centered on artificial intelligence, including machine learning and deep learning.

O-glycosylation site prediction has undergone a great deal of research and advancement, but more work is still needed in this area due to the importance of this task, performance improvement needed as well as the enormous amount of data that is continually being revised. Most of the previous studies for O-glycosylation site prediction use various traditional feature extraction methods to encode protein or peptide sequence information. Sequence-based, structural-based, evolutionary, and multiple sequence alignments (MSAs) are some examples of these approach categories. As the peptide sequence is a sequence of alphabet characters, we here benefit from the embedding of the deep learning language model called Protein language models (PLMs) that exist in the natural language processing (NLP) field. PLMs are transformed-based language models derived from state-of-the-art NLP language models like BERT, ALBERT, and XLNet that are trained on huge protein sequences [6]. The deep learning transformer in PLMs used to capture the contextual information embedded in the amino acids protein sequence. PLMs employed the masked language model that has the ability to create context information around each amino acid and assess its significance in that context [7,8]. Multiple protein language models have been successfully applied for protein sequence embedding and analysis such as ProtBERT, ProtAlbert, ProtXLNet [9], ESM [10], and TAPE [11].

1.2. Literature Review

Previously many computational methods have been applied successfully for O-glycosylation site prediction using machine learning methods. NetOGlyc tool [12] was presented for predicting the mucin-type of O-glycosylation sites by neural networks classifier and using amino acid composition, and surface accessibility features. Oglyc method [13] was built for O-glycosylation site prediction by support vector machine (SVM) method and based on binary profile features and physicochemical properties of protein peptides. Caragea et al. constructed EnsembleGly [14] method for the prediction of O-glycosylation sites using ensembles of SVM that outperformed other classifiers. CKSAAP_OGlySite tool [15] used SVM and "composition of k-spaced amino acid pairs (CKSAAP)" properties for O-glycosylation site prediction. GPP tool was proposed by Hamby [16] to identify O-glycosylation sites using pairwise sequence patterns of amino acid sequences and the random forest (RF) classifier. GlycoEP predictor [3] was developed to identify O-glycosylation sites using multiple classifiers where the SVM model outperformed the other models.

GlycoMine [17] used the random forest classifier to identify O-glycosylation sites using heterogeneous functional and sequence features. O-GlcNAcPRED-II tool [4] was presented for O-glycosylation site prediction using the rotation forest ensembled method that divided the extracted features into four predictors. They implemented the fuzzy under-sampling method (KPCA-FUS) and K-means principal component analysis oversampling approach for creating a balanced dataset. SPRINT-Gly prediction tool [18] by deep neural networks on human and mouse datasets and used various feature extraction techniques. GlycoMine_PU tool [5] was proposed to predict O-glycosylation using positive unlabeled (PU) learning approach. Multiple sequences, functional, and structural-based features were extracted from protein sequences. Zhu et al. developed Captor predictor [19] for o-glycosylation site prediction using SVM and utilizing multiple sequence-based feature extraction methods on the OGP dataset.

In this study, we propose a new method to improve the O-glycosylation site prediction. Firstly, the latest data is collected and preprocessed from the OGP repository. Then, we use the TAPE (Tasks Assessing Protein Embeddings) pre-trained protein language model for feature embedding and representation from the peptide sequences. After that, machine learning algorithms are used to feature selection, model construction, and prediction. Linear SVM is used for selecting the best extracted features followed by the XGBoost ensemble classifier for training and prediction. Finally, cross-validation and independent testing are employed to estimate and compare the proposed model based on MCC, accuracy, sensitivity, specificity, and AUC performance metrics. The paper contributes to developing a new method for O-glycosylation site prediction using machine learning and a pre-trained protein language model. All the previous studies used sequence, structural, and evolutionary-based features extracted from the protein sequences. We here utilize the protein language model for feature embedding instead of the traditional methods for feature extraction. The proposed method achieved high-performance results when compared with the previous recent studies on the same dataset. The remaining of the paper is organized as follows: section 2 describes the materials and methods used in this work including the dataset, protein language model, machine learning method, and evaluation measures. Section 3 presents the experimental performance results and discussion. The last section presents the conclusions and future work.

2. Materials and Methods

2.1. Overview

The General diagram for the proposed method for O-glycosylation site prediction is illustrated in Figure. 1. The figure involves four blocks in which each one representing the main step of the proposed method. The first step represents collecting and preprocessing the used dataset from the OGP repository. The data is split into training and independent datasets. Secondly, the TAPE protein language model is used to extract features from the peptide sequences using embedding. After that machine learning techniques are utilized for feature selection and classification. Linear SVM is implemented for selecting informatic and significant features followed by the XGBoost method for training the proposed model. Finally, cross-validation and independent testing approaches are employed for evaluating and comparing the proposed method based on five performance metrics.

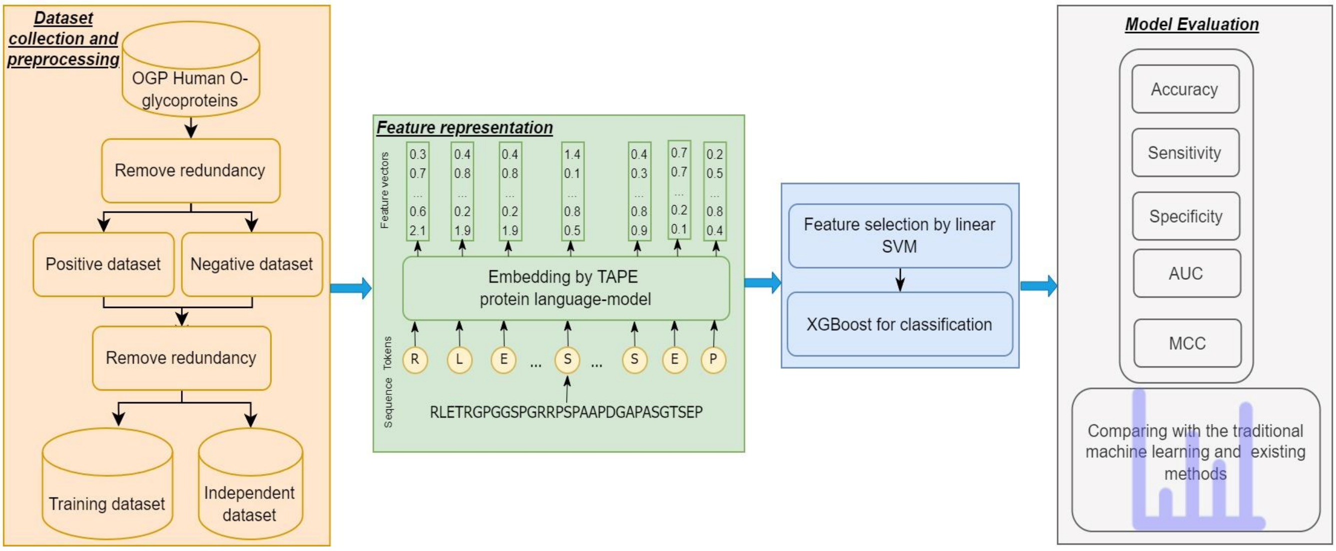


Figure 1. Overall flowchart of the proposed method.

2.2. Dataset Collection and Preprocessing

In this work, the dataset is collected from the O-glycoprotein repository (OGP) [20] (http://www.oglyp.org/download.php). OGP is a database for O-glycosylation sites that contains 2133 O-glycoproteins with 9354 verified O-glycosylation sites. We employ only the human O-glycoproteins that include 1476 glycoproteins with 7038 verified O-glycosylation sites. After that, the CD-HIT program [21] is used to remove redundancy with an identity of over 30%. As result, the used O-glycoproteins are reduced to 1173 O-glycoproteins with 4526 verified O-glycosylation sites. O-glycosylation sites may occur in any serine (S) or threonine (T) amino acids in the glycoproteins sequence. However, not all these sites are considered O-glycosylation sites. We consider the verified O-glycosylation sites acquired from OGP as positive sites or O-glycosylation sites. All the other serine (S) or threonine (T) sites are considered negative sites or non-O-glycosylation sites. The sliding window approach was used for sample construction that divides the sequence into fragments called peptides. Like the Captor study [19] that we compare with, window size 31 is used to construct the O-glycosylation sites in which 15 amino acids downstream and 15 amino acids upstream around the serine (S) or threonine (T) amino acid. Thus, each sample has a length of 31 amino acids. If the length of the sample is less than 31, we extend the sample with the non-known amino acid “X”.

The redundancy samples with a similarity of over 30% are also removed using the CD-HIT program to avoid classification overfitting. The number of positive samples resulting after redundancy removal is 2816 positive samples. To improve prediction performance, 2816 negative samples are selected for constructing a balanced O-glycosylation training dataset. To fairly compare with the recent previous studies, we use the same independent dataset that was used in the Captor study for evaluation and comparison. The independent dataset does not include in the training set and holds 230 positive vs 230 negative samples. Figure 2. shows the window size for each sample with a length of 31 in which the serine (S) or threonine (T) amino acid is at center with 15 residues from its left and 15 residues from its right. The figure also illustrates the two-sample logos [22] of the frequencies of amino acids around positive O-glycosylation sites compared to negative sites using all sample sequences. The figure shows that the Proline (P) amino acids are enriched around the O-glycosylation sites, especially in the sites (-1,

+3, +1, +2, +4, -3, -2). The Alanine (A) amino acids are also enriched around O-glycosylation sites in positions (from -3 to +2). The threonine (T) amino acid is also enriched in position (+1). It is also shown that Leucine (L) is the most depleted amino acid near the O-glycosylation site.

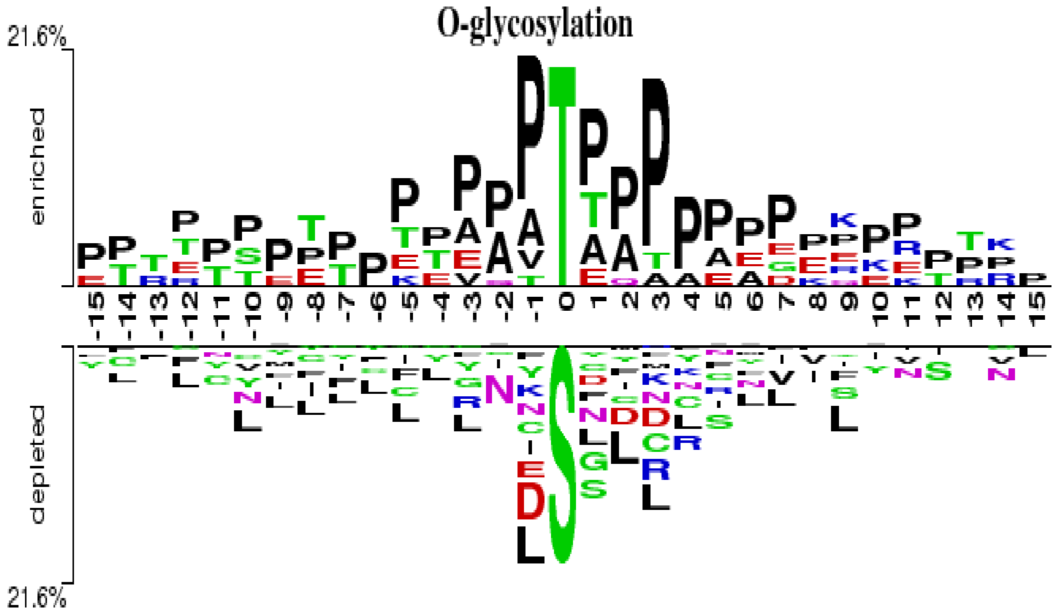


Figure 2. Two-sample logo for the frequencies of amino acids around the O-glycosylation sites.

2.3. Feature Representation

Protein sequences are fragmented into multiple to represent our peptide samples. Twenty symbol letters represent the amino acids of each sample including {A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y}. In addition, the letter 'X' is used to represent unknown amino acids like {U, Z, O, B}. These samples should be encoded to numerical format for training machine learning models. this numerical format represents the key properties and characteristics of the samples. Embedding features extracted from the TAPE [11] protein language model (PLM) from peptide sequences (samples) are used to represent the features. These embedding features contain information about the amino acid conversation in the peptide sequences. The process of TAPE embedding starts by dividing each sample sequence into tokens in which each character (amino acid) is represented by one token. Given the sample $S = \{t_1, t_2, \ldots, t_n\}$ where t_j is a token in the position j in the sample S and n is the sample's length. Embedding function F_{emb} can be represented as:

$$[v_1, v_2, \ldots, v_n] = Femb\ (t_1, t_2, \ldots, t_n) \qquad (1)$$

Where v_j is the features embedded for token t_j which are numeric vectors.

TAPE PLM is based on the BERT masked language model that is calculated by the product of the conditional probabilities of tokens in each site given all other tokens in the sequence by replacing the token at each position with the masked token. This enables conditional non-independence between tokens to be obtained. The masked language model is formulated as:

$$p(t) = \prod_{i}^{n} p(t_i \,| t_1, t_2, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n) \qquad (2)$$

Where t_i is the token in site i and n is the size of the sample sequence.

The TAPE protein language model is trained on a large database called the Pfam database. The Pfam database contains over 31 protein sequences. The last hidden layer in the TAPE transformer is used to extract the embedded features. The length of extracted features for each token is 768. In our case, the length of each sample is 31 tokens. So, the size of extracted futures using TAPE is 768 * 31 = 23,808 features. The properties of the TAPE transformer are shown in Table 1. The table illustrates the key properties of the TAPE transformer including: the number of layers, size of hidden layers, parameters' number, and number of attention heads.

Table 1. Properties of TAPE Transformer.

Property	**Value**
number of layers	12
size of hidden layers	768
number of parameters	92M
number of attention heads	12

2.4. Feature Selection

The extracted features from TAPE embedding are large (23,808 features) that can include noisy and irrelevant features that have a possibly undesirable effect on prediction results. Thus, significant and relevant features are selected by feature selection methods in order to avoid overfitting in the training process as well as to enhance prediction performance. To select the important features, we employed the linear SVM feature selection approach that is comparable with conventional feature selection methods, like information gain and odds ratio [23]. The data sample vector can be represented as $x_i = (x_{i1}, x_{i2}, \dots, x_{in})$ where n is the feature size for each sample. In the linear-kernel SVM, the predictor can be represented:

$$\hat{y} = \sum_{i}^{l} \alpha_i K(x_i, x) + b \qquad (3)$$

For the linear kernel:

$$K(x_i, x) = {x_i}^T . x \qquad (4)$$

The linear-kernel SVM predictor can be rewritten as:

$$\hat{y} = \sum_{j}^{n} w_j x^{(j)} + b \qquad (5)$$

Where b is the bias scaler, α is the initial values for the coefficients, and w_j=$\sum_i^l \alpha_i x_i^{(j)}$. For feature selection, the absolute value $|w_j|$ represents the weight for feature j. The features with the highest coefficient values of $|w_j|$ are selected as optimal features for classification. As the features with low absolute values of w_j have a low impact on the predictions. This indicates that these features are not important for training or classification, and as a result, they could be skipped over in the training stage as well [23]. We used LinearSVC and SelectFromModel functions in the Scikit-learn Python library [24] for feature ranking and selecting the best features. The top 675 features are selected from the 23,808 extracted features by the linear SVM.

2.5. Classification using XGBoost

Extreme gradient boosting (XGBoost) is a state-of-the-art ensemble-based algorithm that is developed for data classification. It has been proved that it outperformed the other traditional classifiers due to its scalability, efficiency, and speed. XGBoost is based on gradient boosting (GB) and decision tree algorithms [25]. Distributed, parallel, and out-of-core computing make XGBoost faster than the other machine learning algorithms. Both GB and XGBoost execute boosting learners using the gradient descent loss technique. GB and XGBoost can be clarified as follows [26]. Given the dataset D=[x,y] where x represents the features and y represents the independent classes. In GB, assuming there are K boosts and B additive functions to predict the results. The $\widehat{y_i}$ represent the prediction for the sample i in boost b and f_b denotes to the tree structure that has weight wj. Then the final prediction for the sample i is represented by:

$$\widehat{y_i} = \sum_{b=1}^{B} f_b(x_i) \qquad (6)$$

The loss of the XGBoost prediction model is minimized by the gradient descent algorithm. XGBoost multiple have hyperparameters that can be adjusted to avoid overfitting and improve performance. The optimal configuration for the XGBoost hyperparameters is clarified in section 3.1. We implement XGBoost by XGBoost python library [25].

2.6. Model Evaluation

Firstly, ten-fold cross-validation is implemented on the training dataset in which the dataset is partitioned randomly into ten folds. Then every fold is selected for the test and the other nine are for training and the result is calculated by the average of the testing result. Secondly, an independent (blind) dataset from the beginning is utilized for independent testing and compares the proposed model with the previous studies. Five common performance metrics are employed involving Matthew's correlation coefficient (MCC), sensitivity, specificity, and accuracy. Moreover, AUC (Area Under the receiver operating characteristic Curve (ROC)) computes the classifier's capability to separate binary data by displaying the true positive rate against the false positive rate. These metrics are represented as:

$$MCC = \frac{tp \times tn - fp \times fn}{\sqrt{(tp + fp)(tp + fn)(tn + fp)(tn + fn)}} \qquad (7)$$

$$\text{Accuracy} = \frac{tp + tn}{tp + fp + tn + fn} \qquad (8)$$

$$\text{Sensitivity} = \frac{tp}{fn + tp} \qquad (9)$$

$$\text{Specificity} = \frac{tn}{fp + tn} \qquad (10)$$

where *tp* denotes the positive sites' number that are truly classified, *fp* (false positive) denotes the positive sites' number that are untruly classified, *tn* denotes the negative sites' number that are truly classified, and *fn* denotes the negative sites' number that are untruly classified.

3. Results and Discussion

3.1. Parameter Setting

XGBoost classifier has various hyperparameters that can be tuned to avoid training overfitting as well as to improve prediction performance. Table 2. shows the XGBoost hyperparameters, their experimented value ranges, and the optimal value used in the model construction.

Table 2. Hyperparameters setting for XGBoost.

Parameter	Value Range	Optimal Value
learning_rate	0 to 1	0.1
max_depth	1 to 10	2
min_child_weight	1 to 10	3
subsample	0 to 1	0.7
booster	gbtree, gblinear	Gbtree

The first tuned parameter is the learning rate that represents the step size shrinkage for overfitting reducing where the default value is 0.3. We tried the values 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9 and the optimal performance was with the learning rate value 0.1. The second parameter is the max_depth which denotes the maximum depth of the tree. If the value of max_depth increases, the model will be prone to overfitting. We tried the integer values between 1 and 10 and the optimal results were with the value 2. The third parameter is the min_child_weight which represents the minimum allowable summation of child weight. The model will be more conservative the greater min child's weight is. We tried the integer values between 1 and 10 and the optimal results were with the value 3. The fourth parameter is the subsample which represents the subsamples of the model before constructing the tree. The XGBoost would randomly sample 50% of the training data before constructing trees if it was set to 0.5. In each boosting cycle, subsampling will take place once. We tried the values 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9 and the optimal performance was with the learning rate value 0.7. The last parameter is the booster which represents the method of boosting in which there are two values (gbtree used for tree-based model and gblinear used for the linear-based model). we used the gbtree booster.

3.2. Evaluation using Cross-validation

Many folds of cross-validation are implemented on the training dataset. In every implementation, data is randomly split into m-fold in which m-1 is employed for training and one for testing. Three-fold, five-fold, eight-fold, and ten-fold cross-validations are implemented. Table 3. Illustrates the results of these implementations. We can observe that the ten-fold cross-validation has the highest performance result among all the implemented cross-validations. The performance results are 0.748 accuracy, 0.7473 sensitivity, 0.7484 AUC, and 0.4962 MCC.

Table 3. Cross-validation performance results on the training dataset.

Fold number	Accuracy	Sensitivity	Specificity	AUC	MCC
three-fold	0.7363	0.7335	0.7401	0.7368	0.4737
five-fold	0.7404	0.7323	0.7493	0.7408	0.4815
eight-fold	0.7383	0.7340	0.7429	0.7384	0.4773
ten-fold	**0.7480**	**0.7473**	**0.7494**	**0.7484**	**0.4962**

From the table, it is observed that the five-fold has the closest performance results to the ten-fold. The ten-fold improved the MCC and sensitivity by 0.01 % over The five-fold cross-validation. The three-fold and eight-fold are lower than the ten-fold by about 0.02% MCC.

3.3. Comparing with Machine Learning Algorithms

XGBoost is compared to five machine learning algorithms: SVM, RF, Linear Regression (LR), Naïve Base (NB), and KNN on the independent dataset. Table 4. illustrates the performance comparison between different machine learning with our proposed method. The comparison shows that XGBoost achieved high-performance results with 0.7761 accuracy, 0.7391 sensitivity, 0.8130 specificity, 0.8295 AUC, and 0.5537 MCC. Figure 3. It is observed that XGBoost outperforms the other traditional machine learning algorithms including SVM, RF, LR, NB, and KNN in terms of accuracy, specificity, AUC, and MCC performance metrics. In terms of sensitivity, KNN outperforms the other classifiers but with low performance with the other metrics. The random forest classifier comes after XGBoost in performance. That is mean that the tree-based classifiers performed better than the other machine learning algorithms.

Table 4. Performance of XGBoost model with SVM, RF, LR, NB, and KNN on the independent dataset.

Classifier	Accuracy	Sensitivity	Specificity	AUC	MCC
SVM	0.7457	0.7609	0.7304	0.8273	0.4915
RF	0.75	0.7348	0.7652	0.8229	0.5002
LR	0.713	0.7391	0.6870	0.7841	0.4267
NB	0.7391	0.6696	0.8087	0.8104	0.483
KNN	0.7196	**0.7696**	0.6696	0.7844	0.4413
XGBoost	**0.7761**	0.7391	**0.8130**	**0.8295**	**0.5537**

3.4. Comparing to the Existing Tools

The proposed method is compared with two recent tools for O-glycosylation site prediction which include Captor [19] and OGP [20] on the independent dataset. for fairly comparison, we used the same independent set that was used in Captor. The performance results of the comparison are shown in

Figure 3. It is clearly found that our proposed method outperforms Captor and OGP in terms of sensitivity, specificity, AUC, and MCC. In terms of accuracy, the Captor exceeds our method by 1%. The proposed method improved the sensitivity by 13% over Captro and 24% over OGP. The proposed method also improved the specificity by 1% over Captro and 2% over OGP. Additionally, the proposed method increased AUC by 12% compared to OGP and by 3% over Captro. The MCC was also improved by 40% compared to OGP and about 30% compared to Captro by the proposed method.

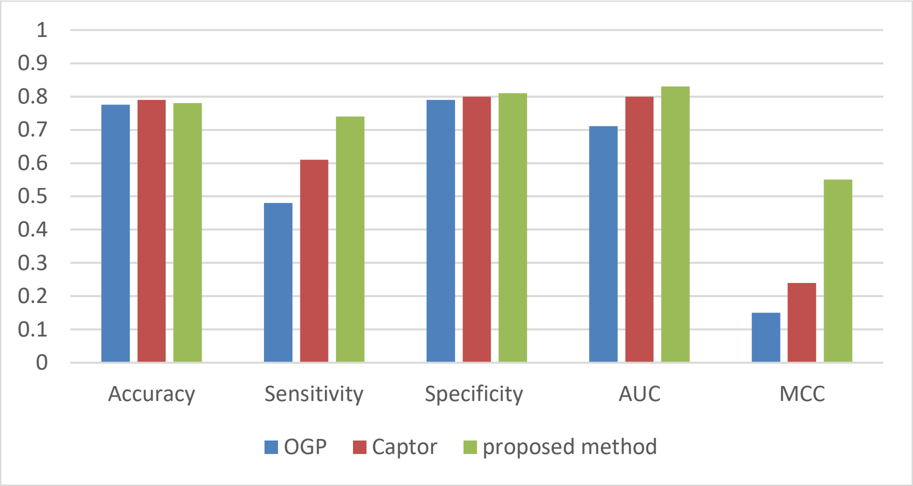


Figure 3. Comparison of performance results of the proposed method against the existing tools on the independent dataset.

4. Conclusions and Future Work

In this paper, we proposed a model for O-glycosylation site prediction using a pre-trained protein language model and machine learning. The dataset is collected from the OGP repository and then it was preprocessed. The TAPE protein language model is employed for feature extraction by embedding strategy. The extracted features are then reduced by the linear SVM method to avoid overfitting and improve performance. XGBoost machine learning method was used for training and classification. The ten-fold cross-validation and independent test were employed to evaluate and validate with accuracy, sensitivity, specificity, AUC, and MCC performance measures. The proposed model was compared with the traditional machine learning methods using the independent dataset in which it outperformed the other machine learning models. On the same independent dataset that is used by the Captro tool, the proposed method was compared to the latest tools for O-glycosylation site prediction including Captor and OGP tools. The comparison results showed that the proposed method outperformed other existing tools. This indicates that features extracted from protein language model embedding perform better than features extracted from traditional feature extraction methods like physicochemical, evolutionary, MSA, or structural-based features. In the future, we plan to use other protein language models and machine learning techniques to predict glycosylation sites.

References

[1] W. He, L. Wei, Q. Zou, Research progress in protein posttranslational modification site prediction, Brief. Funct. Genomics. 18 (2018) 220–229.

[2] A. Alkuhlani, W. Gad, M. Roushdy, A.-B.M. Salem, Intelligent Techniques Analysis for

Glycosylation Site Prediction, Curr. Bioinform. 16 (2021) 774–788.

[3] J.S. Chauhan, A. Rao, G.P.S. Raghava, In silico platform for prediction of N-, O-and C-glycosites in eukaryotic protein sequences, PLoS One. 8 (2013) e67008.

[4] C. Jia, Y. Zuo, Q. Zou, O-GlcNAcPRED-II: an integrated classification algorithm for identifying O-GlcNAcylation sites based on fuzzy undersampling and a K-means PCA oversampling technique, Bioinformatics. 34 (2018) 2029–2036.

[5] F. Li, Y. Zhang, A.W. Purcell, G.I. Webb, K.-C. Chou, T. Lithgow, C. Li, J. Song, Positive-unlabelled learning of glycosylation sites in the human proteome, BMC Bioinformatics. 20 (2019) 112.

[6] C. Marquet, M. Heinzinger, T. Olenyi, C. Dallago, K. Erckert, M. Bernhofer, D. Nechaev, B. Rost, Embeddings from protein language models predict conservation and variant effects, Hum. Genet. (2021).

[7] D. Ofer, N. Brandes, M. Linial, The language of proteins: NLP, machine learning & protein sequences, Comput. Struct. Biotechnol. J. 19 (2021) 1750–1758.

[8] T. Bepler, B. Berger, Learning the protein language: Evolution, structure, and function, Cell Syst. 12 (2021) 654-669.e3.

[9] A. Elnaggar, M. Heinzinger, C. Dallago, G. Rehawi, Y. Wang, L. Jones, T. Gibbs, T. Feher, C. Angerer, M. Steinegger, D. Bhowmik, B. Rost, ProtTrans: Towards Cracking the Language of Life's Code Through Self-Supervised Learning, Ieee Trans Pattern Anal. Mach. Intell. 14 (2021).

[10] A. Rives, J. Meier, T. Sercu, S. Goyal, Z. Lin, J. Liu, D. Guo, M. Ott, C.L. Zitnick, J. Ma, R. Fergus, Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences, Proc. Natl. Acad. Sci. U. S. A. 118 (2021).

[11] R. Rao, N. Bhattacharya, N. Thomas, Y. Duan, X. Chen, J. Canny, P. Abbeel, Y.S. Song, Evaluating protein transfer learning with TAPE, Adv. Neural Inf. Process. Syst. 32 (2019) 1–20.

[12] K. Julenius, A. Mølgaard, R. Gupta, S. Brunak, Prediction, conservation analysis, and structural characterization of mammalian mucin-type O-glycosylation sites, Glycobiology. 15 (2005) 153–164.

[13] S. Li, B. Liu, R. Zeng, Y. Cai, Y. Li, Predicting O-glycosylation sites in mammalian proteins by using SVMs, Comput. Biol. Chem. 30 (2006) 203–208.

[14] C. Caragea, J. Sinapov, A. Silvescu, D. Dobbs, V. Honavar, Glycosylation site prediction using ensembles of Support Vector Machine classifiers, BMC Bioinformatics. 8 (2007) 438.

[15] Y.-Z. Chen, Y.-R. Tang, Z.-Y. Sheng, Z. Zhang, Prediction of mucin-type O-glycosylation sites in mammalian proteins using the composition of k-spaced amino acid pairs, BMC Bioinformatics. 9 (2008) 101.

[16] S.E. Hamby, J.D. Hirst, Prediction of glycosylation sites using random forests, BMC Bioinformatics. 9 (2008) 500.

[17] F. Li, C. Li, M. Wang, G.I. Webb, Y. Zhang, J.C. Whisstock, J. Song, GlycoMine: a machine learning-based approach for predicting N-, C-and O-linked glycosylation in the human proteome, Bioinformatics. 31 (2015) 1411–1419.

[18] G. Taherzadeh, A. Dehzangi, M. Golchin, Y. Zhou, M.P. Campbell, SPRINT-Gly: Predicting N-

and O-linked glycosylation sites of human and mouse proteins by using sequence and predicted structural properties, Bioinformatics. 35 (2019) 4140–4146.

[19] Y. Zhu, S. Yin, J. Zheng, Y. Shi, C. Jia, O-glycosylation site prediction for Homo sapiens by combining properties and sequence features with support vector machine, J. Bioinform. Comput. Biol. 20 (2022) 2150029.

[20] J. Huang, M. Wu, Y. Zhang, S. Kong, M. Liu, B. Jiang, P. Yang, W. Cao, OGP: a repository of experimentally characterized O-Glycoproteins to facilitate studies on O-Glycosylation, Genomics, Proteomics \& Bioinforma. 19 (2021) 611–618.

[21] L. Fu, B. Niu, Z. Zhu, S. Wu, W. Li, CD-HIT: accelerated for clustering the next-generation sequencing data, Bioinformatics. 28 (2012) 3150–3152.

[22] V. Vacic, L.M. Iakoucheva, P. Radivojac, Two Sample Logo: a graphical representation of the differences between two sets of sequence alignments, Bioinformatics. 22 (2006) 1536–1537.

[23] J. Brank, M. Grobelnik, N. Milic-Frayling, D. Mladenic, Feature selection using support vector machines, WIT Trans. Inf. Commun. Technol. 28 (2002).

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, others, Scikit-learn: Machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.

[25] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proc. 22nd Acm Sigkdd Int. Conf. Knowl. Discov. Data Min., 2016: pp. 785–794.

[26] M.M. Bassiouni, I. Hegazy, N. Rizk, E.-S.A. El-Dahshan, A.M. Salem, deep learning approach based on transfer learning with different classifiers for ecg diagnosis, Int. J. Intell. Comput. Inf. Sci. 22 (2022) 44–62.