PreDSLpmo V2.0: A deep learning-based prediction tool for functional annotation of lytic polysaccharide monooxygenases

Kavin S Arulselvan[1$], Priyadharshini Pulavendran[1$], Vaishnavi Saravanan[1$], Ragothaman M. Yennamalli[1*]

1 Department of Bioinformatics, School of Chemical and Biotechnology, SASTRA Deemed to be University, Thanjavur, Tamil Nadu, India 613401.

$ Equal contribution

ORCID:

Ragothaman M. Yennamalli: 0000-0002-3327-1582

Corresponding Author

Dr. Ragothaman M. Yennamalli,

202, ASK2, Department of Bioinformatics,

School of Chemical and Biotechnology,

SASTRA Deemed to be University,

Thanjavur, Tamil Nadu,

India 613401

Ph: +91-8939321002

Email: ragothaman@scbt.sastra.edu

ABSTRACT

Lytic polysaccharide monooxygenases (LPMOs) are redox enzymes in the oxidative cleavage of recalcitrant polysaccharides, facilitating efficient biomass deconstruction for biofuel production. While computational annotation tools have accelerated LPMO discovery, the existing model is limited to the AA9 and AA10 families, leaving the broader diversity of LPMOs underexplored. In this study, we present an improved machine learning framework designed for classifying and annotating LPMOs across the eight families (AA9, AA10, AA11, AA13, AA14, AA15, AA16, and AA17). We curated a high-quality dataset by integrating sequences from GenBank, Joint Genome Institute, and the CAZy database, followed by data cleaning and redundancy reduction using CD-HIT, consisting of 33691 LPMO sequences. Feature extraction was performed using both Python (iFeature) and R-based pipelines, generating over 13,526 descriptors per sequence to capture compositional, physicochemical, and structural properties. Ensemble feature selection was used to identify the significant features for both binary and multiclass classification. An independent dataset was used to estimate the trained models' performance. To address data imbalance, we maintained a 1:1 ratio for all positive and negative sets during both training and validation. A range of machine learning models were systematically trained and evaluated. The multiclass Bi-LSTM model demonstrated higher accuracy, robustness, and generalizability, outperforming traditional approaches. We compared the performance with dbCAN3, a CAZyme predictor. The final model was deployed as a user-friendly web server (https://predlpmo.in), enabling high-throughput, sequence-based functional annotation of LPMOs for the research community. This work bridges a critical gap in LPMO annotation, providing a scalable and reliable solution for enzyme discovery in bioenergy and industrial biotechnology.

Keywords: *LPMO, machine learning, deep learning, Bi-LSTM, Lytic Polysaccharide Monooxygenases, CAZymes.*

INTRODUCTION

Biofuels are increasingly recognized as sustainable energy sources that can help meet the rising energy demands of developing regions while reducing dependence on fossil fuels and mitigating environmental impacts. The efficient deconstruction of plant biomass, especially the conversion of crystalline polysaccharides such as cellulose into fermentable sugars, is central to biofuel production. However, the inherent stability of these polymers against conventional enzymatic and physicochemical treatments presents a significant bottleneck for their industrial utilization, making the development of more effective biocatalysts a priority for the field [1].

LPMOs are copper-dependent enzymes that catalyze the oxidative cleavage of polysaccharides such as cellulose and chitin [2]. By catalyzing the oxidative cleavage of glycosidic bonds in crystalline polysaccharides, LPMOs enhance the accessibility of these substrates to hydrolytic enzymes and boost overall saccharification efficiency. Their ability to boost enzymatic hydrolysis has made them a focus of industrial and biotechnological research. Despite their significance, traditional methods for identifying LPMOs are time-intensive and laborious [3, 4]. This is compounded by the fact that LPMOs share low sequence similarity [5]. However, the active site also known as the catalytic triad is highly conserved [6]. Although, CAZy implements a semi-automated method of identification of LPMOs using a sequence motif and experimental validated published literature [7], this method remains a limitation to annotate family-level LPMO sequences [8]. Initially classified as CBM33 in bacteria and GH61 in fungi, LPMOs are now grouped under the auxiliary activity (AA) class in the CAZy database and further sub-classified into several families based on sequence and substrate specificity. The sequence diversity of LPMOs has expanded rapidly, with new members identified across bacteria, fungi, viruses, and eukaryotes, underscoring the need for accurate and scalable annotation tools [2].

While experimental methods for LPMO identification are valuable, they are labor-intensive and not scalable for the vast influx of new sequence data generated by modern genomics. Computational resources such as dbCAN2 and dbCAN3 have enabled automated CAZyme annotation using HMM profiles and other algorithms, but their specificity for LPMO family classification remains limited, especially as new families and sequence variants are discovered [9].

To address these challenges, a tool called PreDSLpmo was developed, employing both traditional neural networks based on extracted sequence features and bi-directional long short-term memory (Bi-LSTM) models that learn directly from sequence data [10]. These approaches demonstrated improved predictive power over existing tools such as dbCAN2,

particularly for classifying sequences into major LPMO families like AA9 and AA10. However, PreDSLpmo was limited in scope, by focusing mainly on two families – AA9 and AA10; and requiring command-line execution for prediction.

In this study, we present PreDSLpmo v2.0, which significantly expands the classification scope to all existing eight LPMO families and introduces a web-server to overcome the existing limitations. PreDSLpmo v2.0 incorporates both family-specific and multi-class models, leveraging advanced architectures such as Bi-LSTM and feature-based neural networks, and is trained on rigorously curated datasets with redundancy reduction and quality filtering. Importantly, PreDSLpmo v2.0 is deployed as a user-friendly web platform, enabling high-throughput and accessible LPMO annotation for the broader research community. This improved tool addresses the limitations of previous approach and provides a scalable, accurate, and accessible solution for functional annotation of LPMOs in the context of biofuel research and industrial biotechnology.

RESULTS AND DISCUSSION

Dataset Curation

We obtained sequences from NCBI and JGI database based on the entries given in CAZy. The total number of sequences downloaded for each family was as follows: AA1, AA2, AA3, AA4, AA5, AA6, AA7, AA8, AA9, AA10, AA11, AA12, AA13, AA14, AA15, AA16, and AA17 had 21,325; 6,022; 30,124; 1,276; 6,023; 3,285; 2,040; 2,705; 15,066; 12,332; 3,455; 1,518; 413; 1,392; 735; 1,093; and 157 sequences, respectively (Accessed on 6th January 2025). The CAZy database contained entries with identical sequence IDs and organisms for the JGI data, leading to redundancy. Thus, these duplicate entries were removed. Sequences containing ambiguous residues ('X') were also eliminated, and those labeled as partial in the header were excluded to maintain dataset integrity.

After pre-processing, the AA1, AA2, AA3, AA4, AA5, AA6, AA7, AA8, AA9, AA10, AA11, AA12, AA13, AA14, AA15, AA16, and AA17 families had 17,259; 5,627; 29,374; 1,273; 5,973; 3,266; 2,030; 2,689; 14,970; 11,579; 3,442; 1,513; 403; 1,388; 673; 1,090; and 146 sequences, respectively.

The removal of noisy sequences from both the positive and negative sets significantly reduced the total number of sequences in each family. To remove redundancy and ensure more balanced training and validation sets, we applied CD-HIT with a 70 % sequence identity threshold to all families. This step further reduced the sequence counts across the dataset. After CD-HIT filtering, the number of sequences that remained for AA1, AA2, AA3, AA4, AA5, AA6, AA7, AA8, AA9, AA10, AA11, AA12, AA13, AA14, AA15, AA16, and AA17 ranged between. 11.5 % to 51 %.

Since LPMOs are currently classified into eight AA families by CAZy, we considered the machine learning-based model development for all the eight families ranging from AA9, AA10, AA11, AA13, AA14, AA15, AA16, to AA17. Also, the current quantum of sequence data available also The resulting sequences from both the cleaned and CD-HIT-filtered datasets were carefully checked and curated to ensure suitability for machine learning training. To optimize model training, we divided the data into training and test sets using a 60:40 ratio. Furthermore, to prevent potential biases from imbalanced data skewing the model's predictive performance, we maintained a near 1:1 ratio between positive and negative samples during training set construction (Table 1).

Table 1: Data splitting of curated dataset for multi-class model approach.

Positive set (LPMO)			Negative set (Non LPMO)		
Family	**Training**	**Validation**	**Family**	**Training**	**Validation**
AA9	8982	5988	**AA1**	5057	3371
AA10	6947	4632	**AA2**	1648	1099
AA11	2065	1377	**AA3**	8606	5737
AA13	242	161	**AA4**	373	248
AA14	833	555	**AA5**	1750	1166
AA15	404	269	**AA6**	956	638
AA16	654	436	**AA7**	595	396
AA17	88	58	**AA8**	787	525
			AA12	443	296
Total	**20215**	**13476**	**Total**	**20215**	**13476**

Feature identification and selection

Feature sets for all families were derived using the iFeature tool in Python [11], and we generated R scripts utilizing peptides tool (https://cran.r-project.org/web/packages/Peptides/index.html). Using iFeature we generated 21 descriptor sets and 13,494 features per sequence, and using peptides tool we generated 6 features for each sequence resulting in a total of 13500 features (Supplementary Table 1).

Among these features, we proceeded to select significant features by implementing two seperate ensemble feature selection approaches using Python code for (1) binary and for (2) multiclass models, separately. Both approaches assigned values to each descriptors ranging from 0.1 to 1, where a value closer to 1 is highly significant for a descriptor. We set two thresholds (0.5 and 0.6) to identify the minimal set of features to train a model. The features with values of 0.5 and above were selected. At the 0.5 threshold, none of the features generated using the peptides tool met the criteria. Hence, we selected features derived from the iFeature tool that were above the 0.5 threshold value. The selected feature sets included: composition of k-spaced amino acid pairs (CKSAAP, 2,400 descriptors), tripeptide composition (TPC, 8,000 descriptors), Amino acid composition (AAC, 20 descriptors), Distribution (CTDD, 195 descriptors) and Pseudo – amino acid composition (PAAC, 50 descriptors). These features were selected across the families.

Binary Model Performance

The features selected were then used as input into various machine learning algorithms, such as stochastic gradient descent (SGD) classifiers with log loss, hinge, and modified Huber, as well as neural networks (NN). We used GridSearchCV(), a Python function that implements the “fit” and “score” methods, to exhaustively search among various parameters and tune the classifiers for SGD and NN techniques. As noted in previous studies, feature generation methods can sometimes lead to a loss of critical information [12, 13]. To overcome this limitation, we implemented bidirectional long short-term memory (Bi-LSTM) models, which can directly learn abstract sequence patterns from raw protein sequences and are well-suited for capturing long-range dependencies. The capability of LSTMs to retain information from more than 1,000-time steps has led to their wide application [14, 15]. All binary classification models were trained and evaluated using both cleaned datasets and datasets filtered with CD-HIT at a 70 % sequence identity threshold to evaluate the effect of redundancy reduction.

Multiclass Model Performance

In addition to developing binary classifiers, we implemented a multi-class classification strategy in which a single model was trained to differentiate among all eight LPMO families as separate classes. For this multi-class approach, we compared feature-based machine learning methods, such as stochastic gradient descent (SGD) as well as deep learning architectures like recurrent neural networks (RNN) and bidirectional LSTMs (Bi-LSTM). These models were trained and tested using both the cleaned and CD-HIT-non redundant datasets. This multiple approach enabled us to systematically compare the performance of feature-based and deep learning approaches across both binary and multi-class classification tasks, and to assess the impact of redundancy reduction on the robustness of classifiers for LPMO family prediction. The evaluation metrics of multiclass models is listed in Table 2.

The Bi-LSTM model achieved high accuracy and F1-scores for LPMO family classification, with validation F1-scores ranging from 0.9905 to 0.9982 and independent set F1-scores from 0.9952 to 1.0 for most families. Its performance is further illustrated by the confusion matrix (Supplementary Figure 1), which highlights the model’s consistent and robust predictions across all classes. In contrast, dbCAN3 showed more variability and lower precision or recall for some families, as indicated by the comparative metrics. The Bi-LSTM model’s overall accuracy was 98% on the validation set and 99% on the independent set, with an ROC AUC of 0.9965 and average precision of 0.9779. These results indicate that the Bi-LSTM approach provides more consistent and reliable predictions for LPMO family annotation compared to dbCAN3, especially on independent datasets.

Table 2: Comparison of the evaluation metrics of Multiclass approach of Models SGD, Feature-based neural network, Bi-LSTM and dbCAN3 on validation and independent set.

Family	Approach		Stochastic Gradient Descent			Feature-based neural network	Recurrent Neural Network	dbCAN3
	Variant		Hinge	Log	Modified Huber		Bi-Long short-term memory	HMMer, DIAMOND
AA9	Validation Set	True Positive	5312	5158	5229	4600	5854	5846
		False Positive	1095	626	597	607	251	0
		False Negative	676	830	759	1388	134	49
		True Negative	19869	20338	20362	20457	20711	0
		Recall	0.89	0.86	0.87	0.9372	0.977622	0.9917
		Precision	0.83	0.91	0.9	0.7682	0.958886	1
		F1-score	0.86	0.89	0.89	0.8443	0.968163	0.9958
	Independent Set	True Positive	104	100	101	91	114	112
		False Positive	114	19	29	6	7	0
		False Negative	10	14	13	23	0	1
		True Negative	1356	1451	1441	1464	1462	0
		Recall	0.91	0.88	0.89	0.9381	0.9421	0.9912
		Precision	0.48	0.84	0.78	0.7982	1	1
		F1-score	0.63	0.86	0.83	0.8626	0.9702	0.9956

AA10	Validation Set	True Positive	2480	2735	2500	4131	4434	4592
		False Positive	1314	1837	1665	2748	50	0
		False Negative	2152	1897	2132	501	198	25
		True Negative	21006	20383	20650	19572	22268	0
		Recall	0.54	0.59	0.54	0.6005	0.957254	0.9946
		Precision	0.65	0.6	0.6	0.8918	0.988849	1
		F1-score	0.59	0.59	0.57	0.7177	0.972795	0.9973
	Independent Set	True Positive	186	213	189	358	422	414
		False Positive	185	157	150	268	3	0
		False Negative	239	212	236	67	2	4
		True Negative	974	1002	1009	891	1156	0
		Recall	0.44	0.5	0.44	0.5719	0.9929	0.9904
		Precision	0.66	0.58	0.56	0.8424	0.9953	1
		F1-score	0.53	0.54	0.49	0.6813	0.9941	0.9952
AA11	Validation Set	True Positive	1257	1264	1252	1275	1360	1357
		False Positive	244	219	214	94	18	0
		False Negative	120	113	125	102	17	5
		True Negative	25331	25356	25356	25481	25555	0
		Recall	0.91	0.92	0.91	0.9313	0.987654	0.9963

		Precision	0.84	0.86	0.85	0.9259	0.986938	1
		F1-score	0.87	0.89	0.88	0.9286	0.987296	0.9982
	Independent Set	True Positive	41	41	43	43	41	43
		False Positive	15	17	20	3	1	0
		False Negative	2	2	0	0	2	0
		True Negative	1526	1524	1521	1538	1539	0
		Recall	0.95	0.95	1	0.9348	0.9762	1
		Precision	0.73	0.71	0.68	1	0.9535	1
		F1-score	0.83	0.81	0.81	0.9663	0.9647	1
AA13	Validation Set	True Positive	145	144	145	148	154	156
		False Positive	141	17	148	70	2	0
		False Negative	16	17	16	13	7	0
		True Negative	26650	26774	26638	26721	26787	0
		Recall	0.9	0.89	0.9	0.6789	0.956522	1
		Precision	0.51	0.51	0.49	0.9193	0.987179	1
		F1-score	0.65	0.65	0.64	0.781	0.971609	1
	Independent Set	True Positive	1	1	1	2	2	2
		False Positive	2	2	3	0	0	0
		False Negative	1	1	1	0	0	0

		True Negative	1580	1580	1579	1582	1582	0
		Recall	0.5	0.5	0.5	1	1	1
		Precision	0.33	0.33	0.25	1	1	1
		F1-score	0.4	0.4	0.33	1	1	1
AA14	Validation Set	True Positive	497	498	493	514	543	548
		False Positive	303	185	270	111	6	0
		False Negative	58	57	62	41	12	2
		True Negative	26094	26212	26122	26286	26389	0
		Recall	0.9	0.9	0.89	0.8224	0.978378	0.9964
		Precision	0.62	0.67	0.65	0.9261	0.989071	1
		F1-score	0.73	0.77	0.75	0.8712	0.983696	0.9982
	Independent Set	True Positive	19	17	18	21	20	21
		False Positive	20	28	31	26	2	0
		False Negative	4	6	5	2	3	0
		True Negative	1541	1533	1530	1535	1558	0
		Recall	0.83	0.74	0.78	0.4468	0.9091	1
		Precision	0.32	0.38	0.37	0.913	0.8696	1
		F1-score	0.46	0.5	0.5	0.6	0.8889	1
AA15	Validation Set	True Positive	169	168	173	232	246	261

		False Positive	342	101	276	355	16	0
		False Negative	100	101	96	37	22	5
		True Negative	26341	26482	26402	26328	26666	0
		Recall	0.63	0.62	0.64	0.3952	0.91791	0.9812
		Precision	0.33	0.38	0.38	0.8625	0.938931	1
		F1-score	0.43	0.47	0.48	0.5421	0.928302	0.9905
	Independent Set	True Positive	26	26	26	27	29	29
		False Positive	19	15	17	17	1	0
		False Negative	3	3	3	2	0	0
		True Negative	1536	1540	1538	1538	1553	0
		Recall	0.9	0.9	0.9	0.551	0.9667	1
		Precision	0.59	0.63	0.6	0.931	1	1
		F1-score	0.71	0.74	0.72	0.6923	0.9831	1
AA16	Validation Set	True Positive	368	367	369	404	422	427
		False Positive	161	108	212	80	7	0
		False Negative	68	69	67	32	14	2
		True Negative	26355	26308	26299	26436	26507	0
		Recall	0.84	0.84	0.85	0.8347	0.96789	0.9953
		Precision	0.7	0.65	0.64	0.9266	0.983683	1

		F1-score	0.76	0.73	0.73	0.8783	0.975723	0.9977
	Independent Set	True Positive	4	4	5	5	5	5
		False Positive	7	10	9	2	1	0
		False Negative	1	1	0	0	0	0
		True Negative	1572	1569	1570	1577	1577	0
		Recall	0.8	0.8	1	0.625	0.8333	1
		Precision	0.25	0.29	0.36	1	1	1
		F1-score	0.38	0.42	0.53	0.7692	0.9091	1
AA17	Validation Set	True Positive	3	4	0	31	51	57
		False Positive	69	61	156	806	7	0
		False Negative	55	54	53	27	6	0
		True Negative	26825	26733	26738	26088	26886	0
		Recall	0.05	0.07	0.09	0.037	0.894737	1
		Precision	0.04	0.02	0.03	0.5345	0.87931	1
		F1-score	0.05	0.04	0.05	0.0693	0.886957	1
	Independent Set	True Positive	0	0	0	0	0	0
		False Positive	0	0	0	0	0	0
		False Negative	0	0	0	0	0	0
		True Negative	1584	1584	1584	1584	1584	0

		Recall	0	0	0	0	0	0
		Precision	0	0	0	0	0	0
		F1-score	0	0	0	0	0	0

Model Validation

To evaluate the performance of the model, we used an independent dataset of sequences from CAZy. As and when annotations are available sequences are added to CAZY periodically. Thus, the various performance metrics were obtained for this dataset (Table 2). The distribution of the sequences in each family ranged from 2 to 425. AA9, AA10, AA11, AA13, AA14, AA15, AA16, AA17 families demonstrates that dbCAN3 maintains superior classification performance with F1-scores ranging from 0.9958 to 1.0000, while Predlpmo v2.0 approaches exhibit highly variable performance (F1-scores: 0.0693-1.0000), with only family AA14 achieving comparable predictive accuracy to dbCAN3.

The Bi-LSTM model demonstrated excellent predictive performance on the independent test set, achieving an overall accuracy of 99% and a weighted average F1-score of 0.99 (Table 2). Major LPMO families, such as AA9 and AA10, showed near-perfect precision and recall (0.94–1.00 and0.99, respectively), indicating reliable classification for well-represented groups. The model also performed robustly for smaller classes, with AA14 (F1-score 0.89) and AA16 (F1-score 0.91), though performance was somewhat lower for these rare families due to limited sample size. Notably, non-LPMO sequences (AA1, AA2, AA3, AA4, AA5, AA6, AA7, AA8, AA12) were identified with perfect precision and 99% recall, minimizing false positives.

The multiclass Precision-Recall (PR) curves (Figure 1A) illustrate the relationship between precision and recall for each class across varying thresholds, providing insight into the classifier's ability to correctly identify positive instances in the presence of class imbalance. Most classes exhibit high precision across a broad range of recall values, indicating robust positive predictive value and sensitivity. The multiclass Receiver Operating Characteristic (ROC) curves (Figure 1B) further demonstrate the model's discriminative capability, with all classes achieving areas under the curve (AUC) close to or equal to 1.00, signifying excellent separability between classes. The ROC curve, which plots the true positive rate against the false positive rate, confirms that the classifier maintains high sensitivity and specificity across thresholds. Collectively, these results indicate that the model performs exceptionally well in distinguishing among multiple classes, with minimal compromise between precision and recall and negligible misclassification rates. The high AUC and favorable PR curve profiles underscore the reliability and generalizability of the predictive model for multiclass classification tasks.

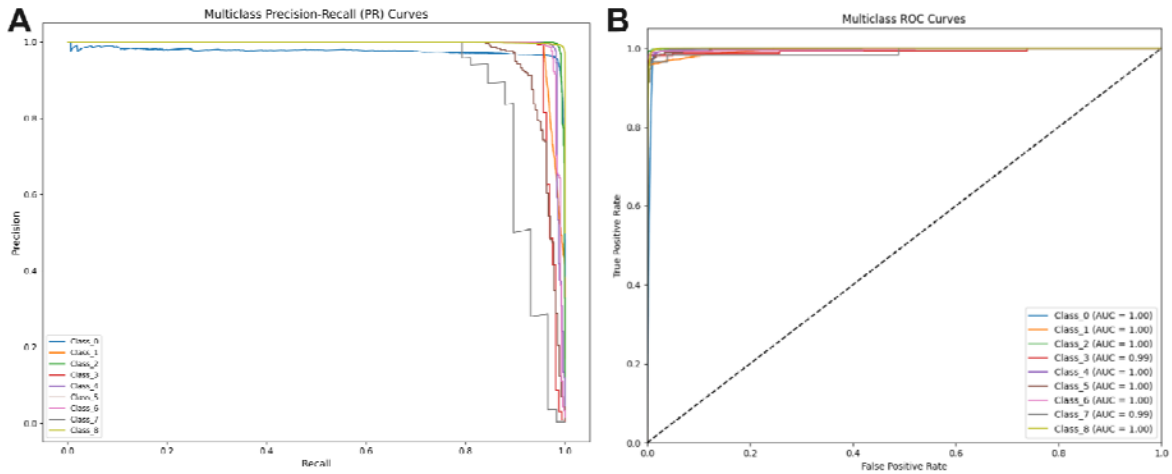

Figure 1. Precision-Recall (PR) and Receiver Operating Characteristic (ROC) Curves of the Bi-LSTM Model. (A) PR curves illustrate the relationship between precision and recall for each class. (B) ROC curves depict the true positive rate versus the false positive rate. In both panels, each colored line represents a different class (Class 0 – AA9, Class 1 – AA10, Class 2 – AA11, Class 3 – AA13, Class 4 – AA14, Class 5 – AA15, Class 6 – AA16, Class 7 – AA17, Class 8 – a composite of AA1, AA2, AA3, AA4, AA5, AA6, AA7, AA8, and AA12) collectively demonstrating the model's discriminative performance across all classes.

PredLpmo Web Server

We have implemented the multi-class trained model as a web server. The preDSLpmo V2.0 web server, called as PredLpmo, addresses the critical need for accessible, high-throughput prediction platforms capable of accurately distinguishing between LPMO families and non-LPMO sequences as depicted in Figure 2. The homepage (Figure 2A) introduces users to the server's deep learning-based prediction capabilities, emphasizing its automated workflow and support for high-throughput LPMO classification and sequence analysis across diverse Auxiliary Activity (AA) families. The submission page (Figure 2B) allows users to input protein sequences either by pasting them in FASTA format or by uploading a file, facilitating flexible and efficient data entry for automated analysis. Upon submission, the results page (Figure 2C) presents the prediction outcomes in a clear tabular format, displaying the sequence ID, predicted family classification as either belonging to a specific AA family or as belonging to Non-LPMO family. For each prediction, the results provide an associated confidence score for each input sequence. Additionally, users can download the results as a CSV file for further analysis, for both individual and high-throughput protein annotation tasks.

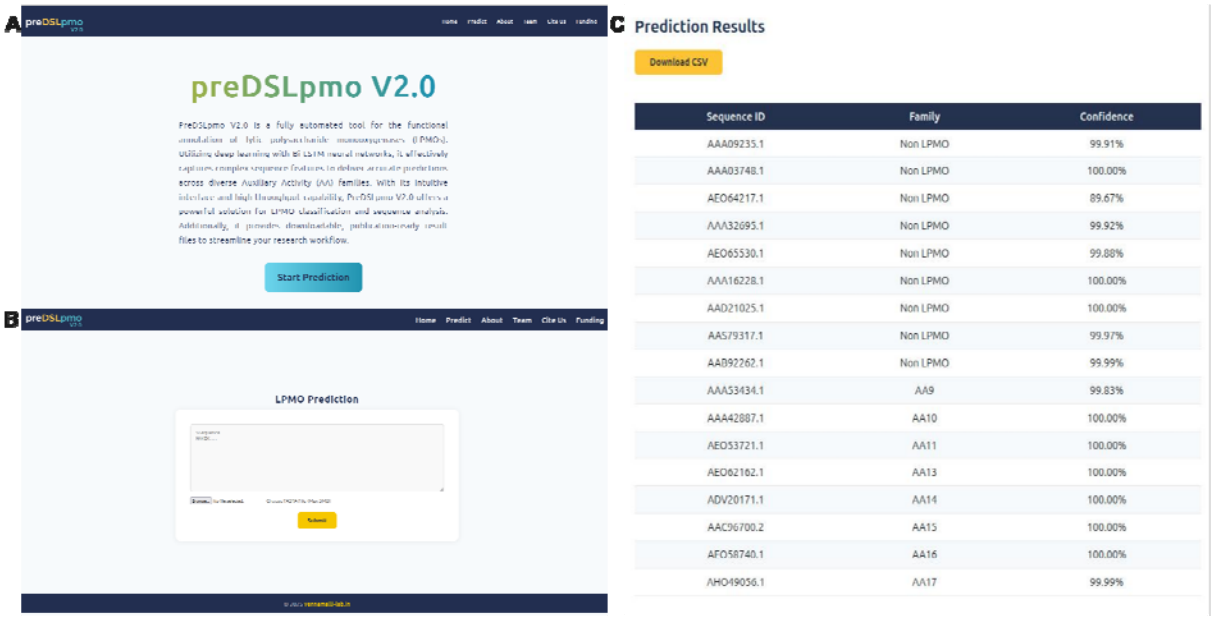

Figure 2: Overview of the preDSLpmo V2.0 Web Server Interface and Workflow. (A) Homepage of the preDSLpmo V2.0 web server, highlighting its deep learning-based prediction capabilities, user-friendly design, and options for high-throughput LPMO classification and sequence analysis. (B) LPMO prediction submission page, where users can input protein sequences in FASTA format directly into the text box or upload a file for automated LPMO classification and analysis. (C) Prediction results page, displaying the sequence ID, predicted family classification (Non LPMO or specific AA family), and the confidence score for each input sequence. Results can be downloaded as a CSV file for further analysis.

METHODOLOGY

To screen and classify LPMO families, we developed a machine learning-based tool that leverages a multiclass prediction strategy for functional annotation of protein sequences. In PreDSLpmo v2.0, sequences are classified into one of eight LPMO families (AA9, AA10, AA11, AA13, AA14, AA15, AA16, AA17) or as non-LPMO (AA1–AA8 and AA12). The workflow, as shown in Figure 3, begins with data extracted from CAZy, NCBI, and JGI databases and preprocessing, comprehensive feature extraction, integrating both iFeature and custom R scripts to generate a total of 13,526 descriptors per sequence, encompassing sequence composition, autocorrelation, physicochemical, and evolutionary features.

LPMO and non-LPMO sequences were downloaded from CAZy (accessed on 6th January, 2025). The LPMO families are AA9, AA10, AA11, AA13, AA14, AA15, AA16, and AA17; and the non-LPMO families are AA1, AA2, AA3, AA4, AA5, AA6, AA7, AA8, and AA12. The corresponding EC numbers for each AA family are listed in Supplementary Table 2.

Sequences were retrieved from two main sources: GenBank and the Joint Genome Institute (JGI). For GenBank, sequences were extracted using accession IDs through the NCBI Entrez API, with a batch retrieval limit of up to 700 sequences per request [16]. These sequences were downloaded in FASTA format. For JGI, the portal was accessed to obtain sequences based on organism names and gene IDs. Initial retrieval involved querying organisms, followed by fetching specific genome sequences corresponding to the identified organisms. A Python script was employed to automate the sequence retrieval process after downloading the organism data. The CAZy database contained repeated entries with the same sequence ID and organism for JGI data, leading to redundancy and duplicates were removed by python script. Sequences containing ambiguous residues ('X') were eliminated, and sequences labeled as partial in the header were excluded using python script to maintain dataset integrity.

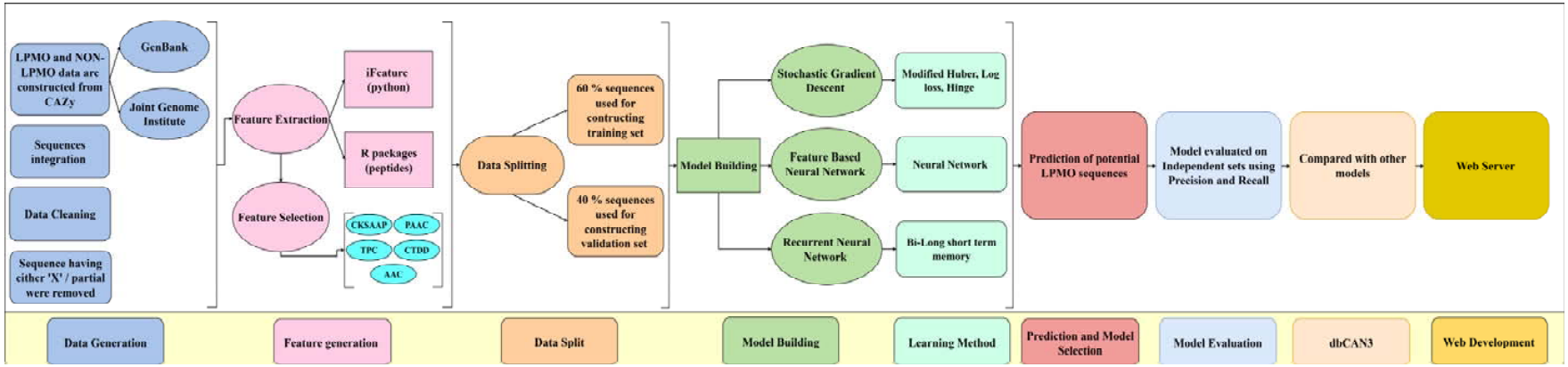

Figure 3: This workflow outlines the complete pipeline for automated LPMO family classification. Sequence data are integrated from GenBank and JGI, cleaned to remove ambiguous and partial entries. Feature extraction is performed using both Python (iFeature) and R packages, followed by feature selection and a 60:40 train-validation split. Multiple machine learning models—including SGD, neural networks, and Bi-LSTM—are trained and evaluated, with the best-performing models deployed as a user-friendly web server for high-throughput LPMO annotation.

Cluster Database at High Identity with Tolerance (CD-HIT) is a widely used bioinformatics tool for clustering and removing redundant protein or nucleotide sequences [17]. We initially used CD-HIT with a 65 % sequence identify for the positive dataset (AA9 to AA17, except AA12, families of LPMOs) and observed that 52 % to 91 % of the sequences were culled, and when the cutoff was kept at 70 %, only 48 % to 86 % of the sequences were culled. Similarly, for the negative dataset, we observed 73 % to 86 % of the sequences were culled at the cutoff of 65 % sequence identity. For the 70 % cutoff, we observed 67 % to 81 % of sequences culled. In order to keep a balanced dataset with a diverse set of sequences we thus decided upon a 70 % sequence identity cutoff to remove redundant sequences. This resulted in this cutoff ensuring that we created a balanced training and validation sets for AA1 to AA17 families Both the cleaned dataset and the CD-HIT-filtered dataset were used for training the model. These corresponding counts are listed in Table 3.

Independent sets for the AA9, AA10, AA11, AA13, AA14, AA15, AA16, and AA17 models were constructed by downloading sequences reported to be AA1, AA2, AA3, AA4, AA5, AA6, AA7, AA8, AA9, AA10, AA11, AA12, AA13, AA14, AA15, AA16 and AA17 by CAZy as of May 8th, 2025. The number of new AA1, AA2, AA3, AA4, AA5, AA6, AA7, AA8, AA9, AA10, AA11, AA12, AA13, AA14, AA15, AA16 and AA17 sequences was 345, 97, 264, 3, 125, 153, 17, 13, 114, 427, 43, 11, 2, 24, 30, 5 and 0 respectively. Independent sets also cleaned and the counts of AA1, AA2, AA3, AA4, AA5, AA6, AA7, AA8, AA9, AA10, AA11, AA12, AA13, AA14, AA15, AA16 and AA17 sequences was 328, 95, 212, 3, 124, 140, 17, 13, 114, 425, 43, 11, 2, 23, 29, 5 and 0 respectively. These corresponding counts are listed in Table 4.

Table 3: Sequence Counts for Each AA Family for training and validation set. The count of sequences downloaded, cleaned, and combined for each AA family from GenBank and JGI databases.

LPMO Family	Count as per CAZy Database (Accessed January 2025)	Redundancy-removed sequences	Genbank Downloaded	JGI Downloaded	Genbank Cleaned	JGI Cleaned	Genbank and JGI Combined	CD Hit (70% identity)	Non - LPMO Family	Count as per CAZy Database (Accessed January 2025)	Redundancy-removed sequences	Genbank Downloaded	JGI Downloaded	Genbank Cleaned	JGI Cleaned	Genbank and JGI Combined	CD Hit(70% Identity)
AA9	15106	15088	1393	13673	1321	13649	14970	3853	**AA1**	21556	21492	8362	12983	4330	12929	17259	4960
AA10	12337	12332	12281	51	11528	51	11579	1336	**AA2**	6062	6055	1324	4698	952	4675	5627	1188
AA11	3482	3480	381	3074	370	3072	3442	1541	**AA3**	30517	30435	4025	26099	3351	26023	29374	9576
AA13	414	414	75	338	65	338	403	56	**AA4**	1275	1276	113	1163	111	1162	1273	247
AA14	1393	1393	143	1249	142	1246	1388	513	**AA5**	6077	6073	1523	4500	1496	4477	5973	1695
AA15	742	735	552	183	490	183	673	346	**AA6**	3315	3292	1185	2100	1172	2094	3266	604
AA16	1094	1093	106	987	104	986	1090	328	**AA7**	2047	2045	209	1831	204	1826	2030	627
AA17	161	157	88	69	77	69	146	75	**AA8**	2710	2708	268	2437	260	2429	2689	714
									AA12	1524	1520	162	1356	160	1353	1513	493

Table 4: **Sequence Counts for Each AA Family for independent set.** The count of sequences downloaded, cleaned, and combined for each AA family from GenBank and JGI databases.

LPMO Family	Count as per CAZy Database (Accessed January 2025)	Redundancy-removed sequences	Newly added sequences in May	Genbank Downloaded	JGI Downloaded	Genbank Cleaned	JGI Cleaned	Genbank and JGI Combined	Non - LPMO Family	Count as per CAZy Database (Accessed January 2025)	Redundancy-removed sequences	Newly added sequences in May	Genbank Downloaded	JGI Downloaded	Genbank Cleaned	JGI Cleaned	Genbank and JGI Combined
AA9	15220	15202	114	25	89	25	89	114	**AA1**	21891	21828	345	290	55	273	55	328
AA10	12737	12729	427	427	0	425	0	425	**AA2**	6143	6140	97	84	13	82	13	95
AA11	3525	3523	43	9	34	9	34	43	**AA3**	30767	30687	264	165	99	113	99	212
AA13	416	416	2	0	2	0	2	2	**AA4**	1278	1278	3	2	1	2	1	3
AA14	1417	1417	24	10	14	9	14	23	**AA5**	6202	6198	125	106	19	105	19	124
AA15	772	765	30	30	0	29	0	29	**AA6**	3468	3444	153	144	9	131	9	140
AA16	1099	1098	5	2	3	2	3	5	**AA7**	2052	2049	17	7	10	7	10	17
AA17	161	157	0	0	0	0	0	0	**AA8**	2718	2716	13	6	7	6	7	13
									AA12	1535	1531	11	4	7	4	7	11

Feature Extraction

To accurately represent the properties of protein sequences, we used a combination of Python and R-based tools for feature extraction. In Python, we used the iFeature package, which allowed us to extract 21 different types of numerical descriptors, resulting in 13,494 features per sequence [11]. These features are mentioned in Supplementary Table 1 covering a wide range of sequence characteristics. This approach provided a detailed, high-dimensional representation of each protein sequence.

In parallel, we developed custom scripts in R using the peptides [18] package to generate six additional features apart from iFeature package for each sequence. These features focused on important biochemical properties such as molecular weight, isoelectric point, hydrophobicity, aliphatic index, instability index, and boman index.

By combining the comprehensive feature extraction capabilities of Python with the specialized biochemical profiling in R, we ensured that our dataset captured both the overall patterns and the specific physicochemical properties of the sequences. This robust representation was essential for accurately distinguishing LPMOs from non-LPMO protein families.

Feature Selection

To optimize feature inputs for machine learning models, we initially implemented ensemble feature selection protocols using R package Ensemble Feature Selection (EFS) [19]. However, due to significant computational limitations-including long runtimes and memory allocation errors-this approach proved impractical for large, high-dimensional datasets. To address these challenges, we re-engineered the feature selection pipeline in Python to replicate and integrate six commonly used methods from R, including the Mann-Whitney U test (median-based), Pearson and Spearman correlations, Logistic Regression, and Random Forest classifiers using both entropy and Gini impurity criteria. Each method independently computed feature importance scores, which were then normalized and combined into a single Ensemble Feature Selection (EFS) score ,enabling improved efficiency and scalability.

For binary classification, we applied an Ensemble Feature Selection (EFS) framework that combined six methods: median-based Mann–Whitney U test, Pearson and Spearman correlation analyses, Logistic Regression coefficients, and Random Forest importance

scores using both Gini and entropy criteria. The feature rankings from these methods were normalized and aggregated into unified EFS scores (ranging from 0 to 1), allowing us to retain only the most discriminative features for each binary task.

For multiclass classification (across nine classes: eight LPMO families and the non-LPMO group), we adapted a modified EFS protocol using multiclass-compatible metrics: Kruskal–Wallis H-test, ANOVA F-statistics, Mutual Information, Multinomial Logistic Regression, and Random Forest importance measures. This approach ensured the selection of features with strong discriminatory power across all classes, minimizing family-specific biases.

By leveraging Python's computational libraries (*scikit-learn*, *scipy*, *numpy*), we achieved substantial improvements in runtime and memory usage-reducing processing time by over 80% compared to the original R-based implementation. This dual strategy ensured robust and scalable feature selection, providing high-quality inputs for both binary and multiclass classification models.

Data Split

To prepare data for model training and evaluation, we divided each dataset into training and validation sets using a 60:40 split. Specifically, 60% of the sequences from both positive (LPMO) and negative (non-LPMO) classes were used for training, while the remaining 40% were set aside for validating. In both sets, we maintained a 1:1 ratio of positive and negative samples to ensure balanced class representation in Table 1.

For binary classification models, we applied ensemble feature selection (EFS) thresholds to filter features before splitting the data. We created separate datasets by selecting features with EFS scores greater than 0.5 and, more stringently, greater than 0.6. This was done for both the cleaned dataset and the CD-HIT clustered dataset.

For the multiclass classification model (covering eight LPMO families and the non-LPMO group), we selected features with EFS scores above 0.5 for both cleaned and CD-HIT datasets. After feature selection, these datasets were also split into training and validation sets using the 60:40 ratio, ensuring balanced representation of all nine classes. This approach allowed us to systematically evaluate the impact of different feature selection thresholds and dataset types on model performance, while maintaining rigorous and unbiased data partitioning.

Model Building

Each Training and validation set for each cleaned and cd-hit dataset were used as input files into the developed Python scripts. The scripts implemented various machine learning algorithms from scikit-learn [20], keras, tensorflow package and machine learning libraries from Python.

Models were developed for both binary and multi-class classification of LPMO families utilizing multiple algorithms, including Stochastic Gradient Descent (SGD) with log loss, hinge loss, and modified huber loss functions; feature-based neural networks; and sequence-based Bi-directional Long Short-Term Memory (Bi-LSTM) networks. For the binary classification task, four distinct models were generated for each LPMO family (AA9, AA10, AA11, AA13, AA14, AA15, AA16, and AA17), resulting in 32 models per algorithm. Accordingly, the SGD algorithm yielded a total of 96 models (32 models each for log loss, hinge loss, and Modified Huber), the feature-based neural network produced 32 models, and the Bi-LSTM architecture generated 16 models due to its sequence-based input modality, in contrast to the feature-based methods. All models were trained on both cleaned and CD-HIT reduced datasets and evaluated using ensemble feature selection (EFS) thresholds of 0.5 and 0.6. Comparative analysis demonstrated that an EFS cutoff of 0.5 consistently delivered optimal performance, which was subsequently adopted for all multi-class classification experiments. Consistent with the binary classification strategy, multi-class models were trained on both cleaned and CD-HIT reduced datasets, with classification labels assigned as follows: class 0—AA9, class 1—AA10, class 2—AA11, class 3—AA13, class 4—AA14, class 5—AA15, class 6—AA16, class 7—AA17, and class 8—non-LPMO sequences for all models.

Stochastic Gradient Descent (SGD)-Based Classification:

Stochastic Gradient Descent (SGD) was employed in this study as an effective optimization algorithm to train linear classifiers on high-dimensional biological sequence data. It iteratively updates model weights to minimize differentiable loss functions, making it especially suitable for large-scale learning tasks. The SGD framework was evaluated using three different loss functions—log loss, hinge loss, and modified Huber loss—to determine their relative effectiveness in classifying Lytic Polysaccharide Monooxygenase (LPMO) sequences.

Log loss corresponds to logistic regression and provides probabilistic predictions, making it appropriate for multi-class classification tasks. Hinge loss, which is the default loss for Support Vector Machines (SVMs), emphasizes maximum-margin classification and is primarily used in binary classification contexts. Modified Huber loss offers a robust, smooth

alternative that combines the benefits of hinge loss and squared loss, making it tolerant to outliers while also yielding probability estimates.

To implement this approach, two separate pipelines were developed. For the multi-class classification task, a single SGD classifier was trained. This model was designed to categorize sequences into one of nine classes—eight representing the LPMO families (AA9 to AA17) and one consolidated non-LPMO class. The dataset was preprocessed through imputation and normalization using SimpleImputer and StandardScaler. These preprocessing components were serialized using Joblib for future reproducibility. Hyperparameter tuning was conducted using GridSearchCV, evaluating a range of values for max_iter (50 to 500) and alpha ($1e^{-7}$ to 1.0) across five-fold cross-validation. Class imbalance was handled by applying class weights and ElasticNet regularization. Early stopping and adaptive learning rates were included to stabilize training and prevent overfitting.

The core training logic resided in the function getBestModel(), which selected the optimal model based on cross-validation results. A heatmap of cross-validation accuracy was generated to visualize performance across hyperparameter combinations. Model evaluation and result generation were handled by saveBestModel(), which produced and saved a comprehensive set of outputs, including confusion matrices, classification reports, ROC curves, and Precision-Recall (PR) curves. For multi-class classification, the script additionally employed label binarization to calculate class-wise ROC and PR metrics. Notably, the final trained model was also evaluated on an independent test set (previously unseen data), ensuring an unbiased assessment of the model's generalizability and robustness.

For the binary classification task, a separate model was built for each LPMO family. The corresponding top-feature dataset for that family is then loaded for both training and validation. As with the multi-class approach, preprocessing steps were applied and saved, and hyperparameters were optimized using grid search. The same three loss functions were explored by adjusting the loss parameter in the SGD classifier. Output evaluation was again performed using saveBestModel(), which saved binary confusion matrices, classification reports, and PR/ROC plots tailored to each family and loss function combination. Just like in the multi-class case, each binary model was also evaluated on an independent test set to validate performance beyond the training data.

This SGD-based classification pipeline provided a highly modular, interpretable, and computationally efficient alternative to deep learning architectures. The ability to swap out

loss functions and apply models across both binary and multi-class tasks made this approach particularly flexible for benchmarking LPMO classification. Additionally, the inclusion of independent test set evaluations strengthened the reliability of the model results and their potential for real-world application in functional annotation workflows.

Feature Based Neural Network

In this study, artificial neural networks were employed to perform both binary and multi-class classification tasks aimed at identifying and classifying lytic polysaccharide monooxygenase (LPMO) sequences into appropriate families. Each model architecture was designed with dense layers and optimized using adaptive optimization techniques to enable accurate learning from high-dimensional sequence-derived features.

To distinguish each LPMO family from non-LPMO sequences, individual binary classifiers were constructed for eight LPMO families (e.g., AA9, AA10, AA11). Each classifier was implemented using a feedforward neural network with four layers: three hidden layers followed by an output layer. The network architecture comprises two hidden layers with 30 neurons each and a third hidden layer with 15 neurons. All hidden layers utilized the Rectified Linear Unit (ReLU) activation function to introduce non-linearity, enhancing the model's capacity to learn complex patterns. The output layer contains two neurons, representing the target LPMO family and the corresponding non-LPMO class, with a softmax activation function to yield class probabilities. The models were trained for 100 epochs with a batch size of 50 using the Adam optimizer and sparse categorical cross-entropy loss function. To evaluate performance, a series of plots and metrics were generated, including training/validation accuracy curves, confusion matrices, Receiver Operating Characteristic (ROC) and Precision-Recall (PR) curves, and a feature correlation heatmap. This approach enabled robust binary discrimination for each family, and models were independently saved along with classification reports and graphical outputs for interpretability and future deployment.

To develop a unified model capable of simultaneously identifying multiple LPMO families, a multi-class classification neural network was constructed. This model was trained to classify input sequences into one of nine classes: eight corresponding to the LPMO families (AA9 to AA17) and one consolidated non-LPMO class. The architecture of this model consisted of four dense layers with ReLU activation in the hidden layers and softmax activation in the output layer. Specifically, the model included layers with 128, 64, and 32 neurons, respectively, and incorporated dropout (0.5 and 0.3) and batch normalization for regularization and improved generalization. The output layer comprised nine neurons, each

representing one class. To address class imbalance in the training dataset, class weights were computed and applied during training. The model was trained for a maximum of 200 epochs with a batch size of 128, incorporating early stopping and learning rate reduction on plateau as callbacks to prevent overfitting and dynamically adjust learning rates.

Comprehensive post-training analyses included confusion matrices, classification reports, and class-wise sample distribution assessments. The training and validation curves for accuracy and loss were also plotted to visualize learning dynamics.

Long short-term memory (Bi-LSTM)

To capture contextual relationships embedded within protein sequences, Bidirectional Long Short-Term Memory (Bi-LSTM) networks were employed due to their proficiency in processing sequential data. Bi-LSTM models leverage both forward and backward sequence information, allowing them to learn comprehensive patterns that are essential for effective classification. In this study, two configurations of Bi-LSTM networks were utilized: binary models trained to identify individual LPMO families, and a unified multi-class model developed to classify sequences across all eight LPMO families and a consolidated non-LPMO class.

For the binary classification task, a dedicated Bi-LSTM model was trained for each LPMO family. Each model began with a character-level tokenization of protein sequences, followed by padding to a fixed sequence length to ensure input uniformity. The architecture consisted of an embedding layer with 300 dimensions, followed by a Bi-LSTM layer comprising 400 units. To enhance generalization and mitigate overfitting, dropout and recurrent dropout mechanisms were applied, both set at a rate of 0.5. The network further included two fully connected layers with 100 and 50 neurons, respectively, using ReLU activation functions, and regularized with L2 penalties. The final layer contained a single neuron activated by a sigmoid function to perform binary classification. The models were trained using the Adam optimizer and binary cross-entropy loss. To account for class imbalance, class weights were dynamically computed based on the distribution of classes in the training data. Training was conducted for up to 100 epochs with a batch size of 128, employing early stopping and adaptive learning rate scheduling to ensure optimal convergence. The performance of each binary model was evaluated using ROC and Precision-Recall curves, confusion matrices, and training history visualizations. This approach enabled high-sensitivity, family-specific predictions, making it well-suited for fine-grained LPMO detection.

In parallel, a multi-class Bi-LSTM model was developed to simultaneously classify sequences into one of nine classes: eight corresponding to individual LPMO families and one representing the non-LPMO group. This model utilized the same character-level sequence representation and padding approach. The network architecture included a 256-dimensional embedding layer, followed by two stacked Bi-LSTM layers with 512 and 256 units, respectively. Regularization was applied using dropout layers and layer normalization to enhance stability and generalization during training. A dense layer with 256 neurons and ReLU activation was employed before the final output layer, which contained nine neurons activated by a softmax function to produce probability distributions across the target classes. The model was trained using sparse categorical cross-entropy and the Adam optimizer, with class weights adjusted to handle imbalanced data. Training was conducted over a maximum of 200 epochs with a batch size of 256, utilizing early stopping, learning rate reduction on plateau, and model checkpointing to retain the best-performing model state. Evaluation of the multi-class model included per-class ROC curve analysis, confusion matrices, classification reports, and plots depicting the evolution of training accuracy and loss over epochs.

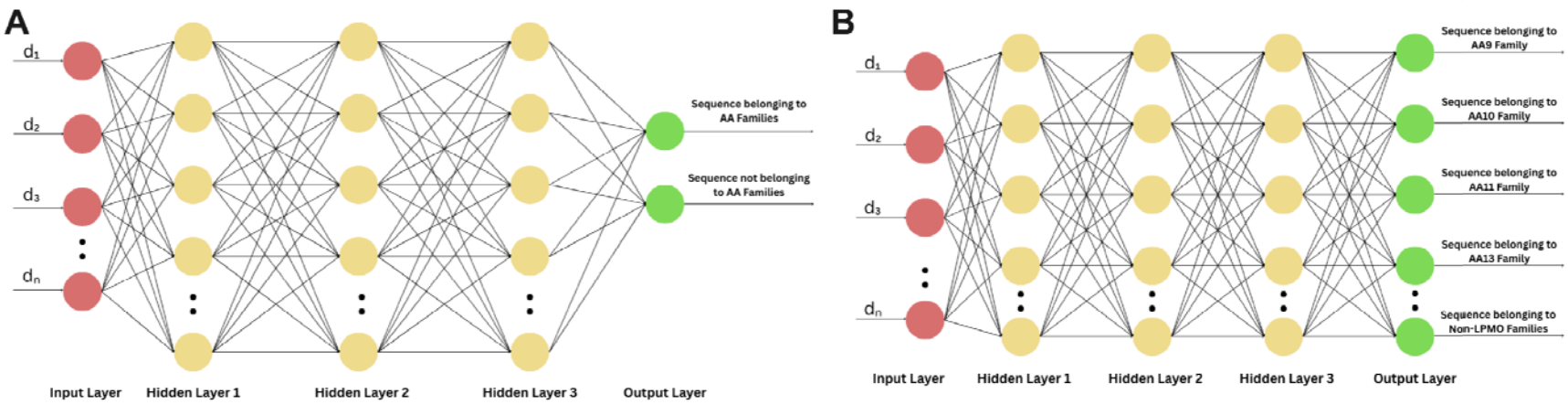

Figure 4: Schematic representation of feature based neural network architectures used for LPMO family classification. (A) Binary neural network model distinguishing AA vs. non-AA sequences. (B) Multiclass neural network model classifying sequences into individual LPMO families and non-LPMO.

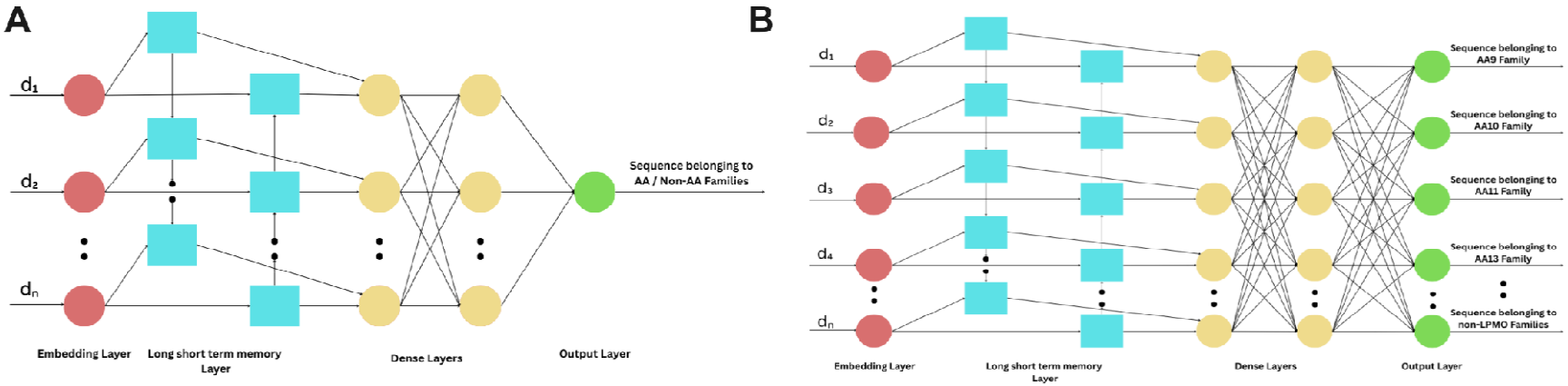

Figure 5: Schematic representation of Bidirectional long short-term memory architectures used for LPMO family classification. (A) Binary Bi-LSTM model classifying AA vs. non-AA sequences. (B) Multiclass Bi-LSTM model differentiating among LPMO families and non-LPMO.

Model Evaluation and Validation

To rigorously evaluate classifier performance, we prioritized precision, recall, and F1-score as primary metrics, complemented by ROC-AUC and confusion matrices [21]. These metrics were selected to address potential class imbalances and ensure robust interpretation of model efficacy.

Precision

$$Precision = \frac{(TP)}{(TP + FP)}$$

Recall

$$Recall = \frac{(TP)}{(TP + FN)}$$

F1-score

$$F_1 = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)}$$

Where, *TP* stands for True Positives, *FP* stands for False Positives, and *FN* stands for False Negatives. For model validation, the performance of the trained Bi-LSTM model was assessed using an independent test set that was not seen during training. This process began by loading the optimized Bi-LSTM model and its corresponding tokenizer. Sequences from the independent dataset were converted into integer-encoded representations using the tokenizer and then padded or truncated to a fixed length of 400 residues to ensure compatibility with the model's input requirements. The model generated predictions for each sequence, providing both class probabilities and the most likely class label across all nine categories (eight LPMO families and non-LPMO). To comprehensively evaluate the model's predictive accuracy, a classification report was generated, detailing precision, recall, and F1-score for each class. Additionally, a confusion matrix was constructed and visualized as a heatmap to illustrate the distribution of true versus predicted labels. All results, including the classification report, confusion matrix, and the predicted class assignments, were saved for further analysis.

dbCAN3

dbCAN3, a meta-server for CAZyme annotation, combines HMMER (dbCAN and dbCAN-sub; E-value < 1×10^{-15}, coverage > 0.35) [22] and DIAMOND (E-value < 1×10^{-102}) to predict CAZyme families and substrates. To benchmark our tool, we submitted the same test and independent datasets to the dbCAN3 server (https://bcb.unl.edu/dbCAN2/blast.php) using its default parameters [23].

Web Server Deployment and User Accessibility

Our web server is a full-stack application that automates the classification of protein sequences into Lytic Polysaccharide Monooxygenase (LPMO) families using machine learning. The backend, built with Node.js, handles HTTP requests, file uploads, and integrates a Python-based ML model for predictions. When users submit sequences through text or FASTA files, the server saves inputs as temporary files, invokes the Python script to load a pre-trained Keras model, processes sequences through tokenization/padding, and returns JSON results with predicted LPMO families (AA9, AA10, AA11, AA13, AA14, AA15, AA16, AA17/Non-LPMO) and confidence scores. The backend also manages error handling, CSV parsing, and cleanup of temporary files.

The frontend, developed in React.js, provides a responsive interface with routes for submission, results display, project background, and team/citation info. Custom CSS ensures modern styling with gradients, shadows, and mobile-responsive layouts. Key interactions include Users input sequences through textarea or file upload on /predict. The frontend sends data to /api/predict, displays loading states, and navigates to /results upon completion. Results are shown in a table with CSV download options, while the About page explains the scientific context of LPMO classification.

CONCLUSION

Through a rigorous computational framework, we successfully collected, analyzed, and functionally annotated lytic polysaccharide monooxygenases (LPMOs) using sequence-based machine learning. To ensure optimal input for modeling, we applied ensemble feature selection for binary tasks and a modified multiclass feature selection protocol for multiclass classification, retaining only the most informative features. For binary modeling, we trained separate models for each family using SGD classifiers with log, hinge, and modified huber loss functions, as well as feature-based neural networks, in addition to sequence-based Bi-LSTM models. For the multiclass approach, we further evaluated all other methods used in binary methods. After rigorous cross-validation and benchmarking with dbCAN3 and all other models, the multiclass Bi-LSTM model demonstrated superior accuracy and robustness, providing clear family assignments and confidence scores for each input sequence. This multiclass approach overcomes the ambiguity of overlapping predictions in binary models, enabling reliable and user-friendly LPMO family classification for large-scale sequence screening. By evaluating diverse models-including SGD variants, feedforward neural networks, and Bi-LSTM architectures-on both cleaned and CD-HIT clustered datasets, we demonstrated that the Bidirectional LSTM (Bi-LSTM) multiclass model achieved superior performance in discriminating LPMO families. This model's ability to capture long-range sequence dependencies and structural motifs enabled accurate classification of sequences into eight LPMO families (AA9–AA17, excluding AA12) alongside a consolidated non-LPMO class. The optimized Bi-LSTM pipeline was subsequently deployed as a publicly accessible web server (https://predlpmo.in) accepting raw protein sequences as input and providing real-time predictions of LPMO family affiliation with confidence scores. This tool streamlines high-throughput functional annotation of LPMOs, bridging sequence data and biochemical characterization for researchers in bioenergy and enzymology.

Funding

RMY is funded by Department of Biotechnology, Ministry of Science and Technology, Government of India (BT/PR42225/BCE/8/1586/2021).

CRediT authorship contribution statement

KSA, PP, and VS did the data collection, data processing, and data curation. They equally contributed towards testing and validating the various models. All three co-wrote the first draft of the manuscript. RMY conceived, supervised and finalized the manuscript.

Declaration of Competing Interest

The authors declare that there are no competing interests.

Acknowledgements

All the authors thank SASTRA Deemed to be University for infrastructural support.

Code Availability

The codes used in this study are available at:
https://github.com/raghuyennamalli/PreDSLpmo_v2 and the webserver is available at:
https://predlpmo.in/

REFERENCES

1. Johansen KS. Discovery and industrial applications of lytic polysaccharide mono-oxygenases. Biochem Soc Trans. 2016 Feb;44(1):143-9. doi: 10.1042/BST20150204. PMID: 26862199.
2. Vandhana TM, Reyre JL, Sushmaa D, Berrin JG, Bissaro B, Madhuprakash J. On the expansion of biological functions of lytic polysaccharide monooxygenases. New Phytol. 2022 Mar;233(6):2380-2396. doi: 10.1111/nph.17921. Epub 2022 Jan 8. PMID: 34918344.
3. Rieder L, Sørlie M. Recent Advances in Understanding LPMO Catalysis. Biochemistry. 2023 Nov 21;62(22):3170-3172. doi: 10.1021/acs.biochem.3c00458. Epub 2023 Nov 2. PMID: 37916429.
4. Wang D, Li Y, Zheng Y, Hsieh YSY. Recent Advances in Screening Methods for the Functional Investigation of Lytic Polysaccharide Monooxygenases. Front Chem. 2021 Apr 12;9:653754. doi: 10.3389/fchem.2021.653754. PMID: 33912540.
5. Forsberg Z, Stepnov AA, Nærdal GK, Klinkenberg G, Eijsink VGH. Engineering lytic polysaccharide monooxygenases (LPMOs). Methods Enzymol. 2020;644:1-34. doi: 10.1016/bs.mie.2020.04.052. Epub 2020 May 15. PMID: 32943141.
6. Book AJ, Yennamalli RM, Takasuka TE, Currie CR, Phillips GN Jr, Fox BG. Evolution of substrate specificity in bacterial AA10 lytic polysaccharide monooxygenases. Biotechnol Biofuels. 2014 Aug 6;7:109. doi: 10.1186/1754-6834-7-109. PMID: 25161697.
7. Levasseur A, Drula E, Lombard V, Coutinho PM, Henrissat B. Expansion of the enzymatic repertoire of the CAZy database to integrate auxiliary redox enzymes. Biotechnol Biofuels. 2013 Mar 21;6(1):41. doi: 10.1186/1754-6834-6-41. PMID: 23514094.
8. Drula E, Garron ML, Dogan S, Lombard V, Henrissat B, Terrapon N. The carbohydrate-active enzyme database: functions and literature. Nucleic Acids Res. 2022 Jan 7;50(D1):D571-D577. doi: 10.1093/nar/gkab1045. PMID: 34850161.
9. Sabbadin F, Hemsworth GR, Ciano L, Henrissat B, Dupree P, Tryfona T, Marques RDS, Sweeney ST, Besser K, Elias L, Pesante G, Li Y, Dowle AA, Bates R, Gomez LD, Simister R, Davies GJ, Walton PH, Bruce NC, McQueen-Mason SJ. An ancient family of lytic polysaccharide monooxygenases with roles in arthropod development and biomass digestion. Nat Commun. 2018 Feb 22;9(1):756. doi: 10.1038/s41467-018-03142-x. PMID: 29472725.
10. Srivastava PA, Hegg EL, Fox BG, Yennamalli RM. PreDSLpmo: A neural network-based prediction tool for functional annotation of lytic polysaccharide monooxygenases. J

Biotechnol. 2020 Jan 20;308:148-155. doi: 10.1016/j.jbiotec.2019.12.002. Epub 2019 Dec 9. PMID: 31830497.

11. Chen Z, Zhao P, Li F, Leier A, Marquez-Lago TT, Wang Y, Webb GI, Smith AI, Daly RJ, Chou KC, Song J. iFeature: a Python package and web server for features extraction and selection from protein and peptide sequences. Bioinformatics. 2018 Jul 15;34(14):2499-2502. doi: 10.1093/bioinformatics/bty140. PMID: 29528364.
12. Ali A, Jillani F, Zaheer R, Karim A, Alharbi YO, Alsaffar M, Alhamazani K. Practically implementation of information loss: sensitivity, risk by different feature selection techniques. IEEE Access. 2022 Feb 18;10:27643-54.
13. Jia, W., Sun, M., Lian, J. et al. Feature dimensionality reduction: a review. Complex Intell. Syst. 8, 2663–2693 (2022). https://doi.org/10.1007/s40747-021-00637-x
14. Neil D, Pfeiffer M, Liu SC. Phased LSTM: Accelerating Recurrent Network Training for Long or Event-based Sequences. arXiv (Cornell University). 2016 Jan 1
15. Yu Y, Si X, Hu C, Zhang J. A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. Neural Comput. 2019 Jul;31(7):1235-1270. doi: 10.1162/neco_a_01199. Epub 2019 May 21. PMID: 31113301.
16. Wheeler DL, Barrett T, Benson DA, Bryant SH, Canese K, Chetvernin V, Church DM, DiCuccio M, Edgar R, Federhen S, Geer LY, Kapustin Y, Khovayko O, Landsman D, Lipman DJ, Madden TL, Maglott DR, Ostell J, Miller V, Pruitt KD, Schuler GD, Sequeira E, Sherry ST, Sirotkin K, Souvorov A, Starchenko G, Tatusov RL, Tatusova TA, Wagner L, Yaschenko E. Database resources of the National Center for Biotechnology Information. Nucleic Acids Res. 2007 Jan;35(Database issue):D5-12. doi: 10.1093/nar/gkl1031. Epub 2006 Dec 14. PMID: 17170002.
17. Li W, Godzik A. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. Bioinformatics. 2006 Jul 1;22(13):1658-9. doi: 10.1093/bioinformatics/btl158. Epub 2006 May 26. PMID: 16731699.
18. Osorio D, Rondón-Villarreal P, Torres R. Peptides: A Package for Data Mining of Antimicrobial Peptides. The R Journal. 2015;7(1):4. https://doi.org/10.32614/RJ-2015-001.
19. Neumann U, Genze N, Heider D. EFS: an ensemble feature selection tool implemented as R-package and web-application. BioData Min. 2017 Jun 27;10:21. doi: 10.1186/s13040-017-0142-8. PMID: 28674556.
20. Pedregosa F, Buitinck L, Louppe G, Grisel O, Varoquaux G, Mueller A. Scikit-learn. GetMobile: Mobile Computing and Communications. 2015 Jun 1;19(1):29–33.
21. Saito T, Rehmsmeier M. The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PLoS One. 2015 Mar 4;10(3):e0118432. doi: 10.1371/journal.pone.0118432. PMID: 25738806.

22. Finn RD, Clements J, Eddy SR. HMMER web server: interactive sequence similarity searching. Nucleic Acids Res. 2011 Jul;39(Web Server issue):W29-37. doi: 10.1093/nar/gkr367. Epub 2011 May 18. PMID: 21593126.
23. Zheng J, Ge Q, Yan Y, Zhang X, Huang L, Yin Y. dbCAN3: automated carbohydrate-active enzyme and substrate annotation. Nucleic Acids Res. 2023 Jul 5;51(W1):W115-W121. doi: 10.1093/nar/gkad328. PMID: 37125649.