DeepBIO is an automated and interpretable deep-learning platform for biological sequence prediction, functional annotation, and visualization analysis

Ruheng Wang[1,2#], Yi Jiang[1,2#], Junru Jin[1,2#], Chenglin Yin[1,2#], Haoqing Yu[1,2], Fengsheng Wang[1,2], Jiuxin Feng[1,2], Ran Su[3], Kenta Nakai[4], Quan Zou[5*], and Leyi Wei[1,2*]

[1]School of Software, Shandong University, Jinan, China;

[2]Joint SDU-NTU Centre for Artificial Intelligence Research (C-FAIR), Shandong University, Jinan, China;

[3]College of Intelligence and Computing, Tianjin University, Tianjin, China;

[4]Human Genome Center, Institute of Medical Science, University of Tokyo, Tokyo, Japan;

[5]Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China.

[#]These authors contributed equally to this work as first authors.

[*]Corresponding authors:

Q.Z: zouquan@nclab.net

L.W: weileyi@sdu.edu.cn

Abstract

Here, we present DeepBIO, the first-of-its-kind automated and interpretable deep-learning platform for biological sequence functional analysis. DeepBIO is a one-stop-shop web service that enables researchers to develop a new deep-learning architecture to answer any biological question. Specifically, given any biological sequence data, DeepBIO supports a total of 42 state-of-the-art deep-learning algorithms for model training, comparison, optimization, and evaluation in a fully automated pipeline. DeepBIO provides a comprehensive result visualization analysis for the predictive models covering several aspects, such as model interpretability, feature analysis, functional sequential region discovery, *etc*. Additionally, DeepBIO supports over 20 functional site annotation tasks using deep-learning architectures, with comprehensive interpretations and graphical visualizations to validate the reliability of annotated sites. Empowered by high-performance computers, DeepBIO allows fast prediction with up to million-scale sequence data, demonstrating its usability in real application scenarios. Case study results show that DeepBIO provides an accurate, robust, and interpretable prediction, demonstrating the power of deep learning in biological sequence analysis. Overall, we expect DeepBIO to ensure the reproducibility of deep-learning biological sequence analysis, lessen the programming and hardware burden for biologists, and provide meaningful functional insights at both sequence-level and base-level from biological sequences alone. DeepBIO is publicly available at http://inner.wei-group.net/DeepBIO.

Introduction

The development of next-generation sequencing techniques has led to an exponential increase in the amount of biological sequence data accessible, including genomic, transcriptomic, and proteomic sequences, *etc*. It naturally poses an important challenge - how to build the relationships from sequences to structures and functions [1]. Given such large-scale data, traditional wet-lab experimental methods are laborious, time-consuming, and high-cost for functional analysis. Alternatively, the booming development of machine learning approaches has paved a new way to understand the complex mapping from biological sequences to their structures and functional mechanisms. Over the past few decades, data-driven machine learning approaches have emerged as powerful methods to enable the automated and fast function prediction ab initio from sequences alone, providing a new perspective on studying the functional aspects of biological sequences [2, 3].

The increasing use of machine learning workflows in biological sequence analysis and the willingness to disseminate trained machine learning models have pushed computer scientists to design more user-friendly solutions. In recent years, there is an increasing number of web servers and software packages developed for this purpose. For instance, iLearnplus is a machine-learning platform that provides an automated pipeline for biological sequence analysis, which comprises feature extraction and selection, model construction, and analysis of prediction results [4]. Liu et al. developed a similar web platform namely, BioSeq-BLM. Besides, other representative tools include Rcpi [5], protr [6], *etc*. The platforms and tools have boosted the use of machine learning solutions for biological sequence analysis tasks in biology.

However, these traditional machine-learning workflows have some drawbacks. For instance, to train a good model, strong professional knowledge is usually required in terms of designing feature descriptors, selecting machine learning algorithms, as well as setting up the model parameters, which limit their usability in real applications to some extent; on the other hand, they cannot support large-scale prediction and analysis. Recently, deep learning has played a complementary role over traditional machine learning due to its excellent scalability and adaptivity. Therefore, some deep-learning analysis tools have been developed, such as Kipoi [7], Pysster [8], and Selene [9]. Kipoi [7] is a repository of ready-to-use trained models for genomic data analysis. Pysster [8] is a Python package that supports training deep-learning models only with convolutional neural networks on biological sequence data. More recently, Chen et al. developed Selene, a PyTorch-based deep-learning library for quick and simple development, training, and application of deep-learning model architectures for biological sequence data [9].

These tools make the use of deep learning more convenient for data-driven biological sequence analysis to some extent. However, there remain some key challenges that need to be addressed. First, despite the availability of deep-learning tools, they are not completely automated. Running the tools still faces some technical challenges for non-expert researchers. On one hand, it requires strong professional knowledge and programming skills to set up sophisticated software, which directly limits the generic use of deep learning in nonexpert communities (i.e., researchers without any computer science training). On the other hand, it is extremely difficult for researchers to train deep-learning models with the enormous scale of data space, since the major downside of deep learning is its computational intensity, requiring high-performance computational resources and long training time. Accordingly, a web platform that enables an automated deep-learning pipeline is highly required. Second, most of the existing tools cannot meet the high demand of the research community, since they provide few deep-learning architectures, such as convolutional neural networks, for model construction. In fact, there are some state-of-the-art deep-learning models with successful applications in bioinformatics problems. For example, Graph Neural Networks demonstrated their ability to address complex biological problems, for example, the prediction of microRNA-disease interaction [10]. Large-scale language pre-trained models like DNABERT and ProtBERT, showed strong capacities in discriminative feature learning in biological sequence analysis [2, 11]. Thus, it is necessary to enable the use of state-of-the-art deep-learning approaches in biology. Ultimately, existing tools lack comprehensive result analysis, which might limit users' understanding of what the deep-learning models learn; how reliable the deep-learning predictions are; and why the deep learning performs well.

In this work, we present DeepBIO, an automated deep-learning platform for biological sequence prediction, functional annotation, and result visualization analysis. To be specific, our DeepBIO distinguishes itself from other platforms with the following unique advantages: **(1) One-stop-shop service.** DeepBIO is the first-of-its-kind platform as a code-free web portal to ensure the reproducibility of deep-learning biological sequence analysis and to lessen the programming burden for biologists. **(2) Pure deep-learning platform.** DeepBIO is a purely deep-learning platform that integrates over 40 state-of-the-art mainstream deep-learning

algorithms, including convolutional neural networks, advanced natural language processing models, and graph neural networks, which enables researchers of interest to train, compare, and evaluate different architectures on any biological sequence data. **(3) Two predictive modules.** DeepBIO is the first platform that supports not only sequence-level function prediction for any biological sequence data, but also allows over 20 base-level functional annotation tasks using pre-trained deep-learning architectures, covering DNA methylation, RNA methylation, and protein-binding specificity. Besides, we further offer an in-depth comparative analysis between the predictions by our models and by experimental data to validate the reliability of the predictions. Notably, empowered by high-performance computers, we demonstrate that DeepBIO supports fast prediction with up to million-scale sequence data, demonstrating its usability in real application scenarios. **(4) Comprehensive result visualization analysis**. Aiming to help researchers to dig into more insights from the model predictions, DeepBIO offers a comprehensive visualization result analysis with a variety of interactive figures and tables, covering the following five aspects: statistics analysis of input data; prediction models' performance evaluation and comparison; feature importance and interpretable analysis; model parameter analysis; and sequence conservation analysis. In particular, integrating interpretable mechanisms (*e.g.*, attention heatmap, and motif discovery) into deep-learning frameworks enables researchers to analyze which sequential regions are important for the predictions, addressing the issue of the "black box" in deep learning and effectively building the relationship between biological sequences to functions.

MATERIALS AND METHODS

The overall framework of DeepBIO

The DeepBIO platform fully automates the model training process and applies more than forty deep-learning approaches for sequence classification prediction and functional site analysis based on genomic and proteomic data. **Figure 1A** illustrates the overall framework of the proposed DeepBIO, which consists of four modules: (1) Data input module, (2) Deep-learning prediction module, (3) Sequence functional annotation module, and (4) Result report module. The workflow of DeepBIO is described as follows. Firstly, DeepBIO takes biological sequence data from the data input module, where we can handle three biological sequence types including DNA, RNA, and protein. Next, there are two functional modules for two types of tasks respectively; one is the deep-learning prediction module for binary classification tasks, and the other is the sequence functional annotation module for functional annotation tasks. The detailed web techniques and the overview of the interaction between the front end and back end are shown in **Figure 1B**. In the deep-learning prediction module, we support users in automatically training, evaluating, and comparing the deep-learning models with their input data. Specifically, there are four main steps (seen in **Figure 1C**) in this module: (I) Data pre-processing, (II) Model construction, (III) Model evaluation, and (IV) Visualization analysis. In the functional annotation module, we provide the base-level functional annotation using deep-learning approaches, such as DNA methylation, RNA methylation, and protein-binding specificity prediction. Similarly, we also curated four steps for this module (seen in **Figure 1C**): (I) Data selection, (II) Task selection, (III) Model loading, and (IV) Result visualization. Ultimately, for the convenience of users to download the prediction results, we provide

abundant analysis results involved with various kinds of data formats in the result report module. In the following, we emphasize the details of these four modules.

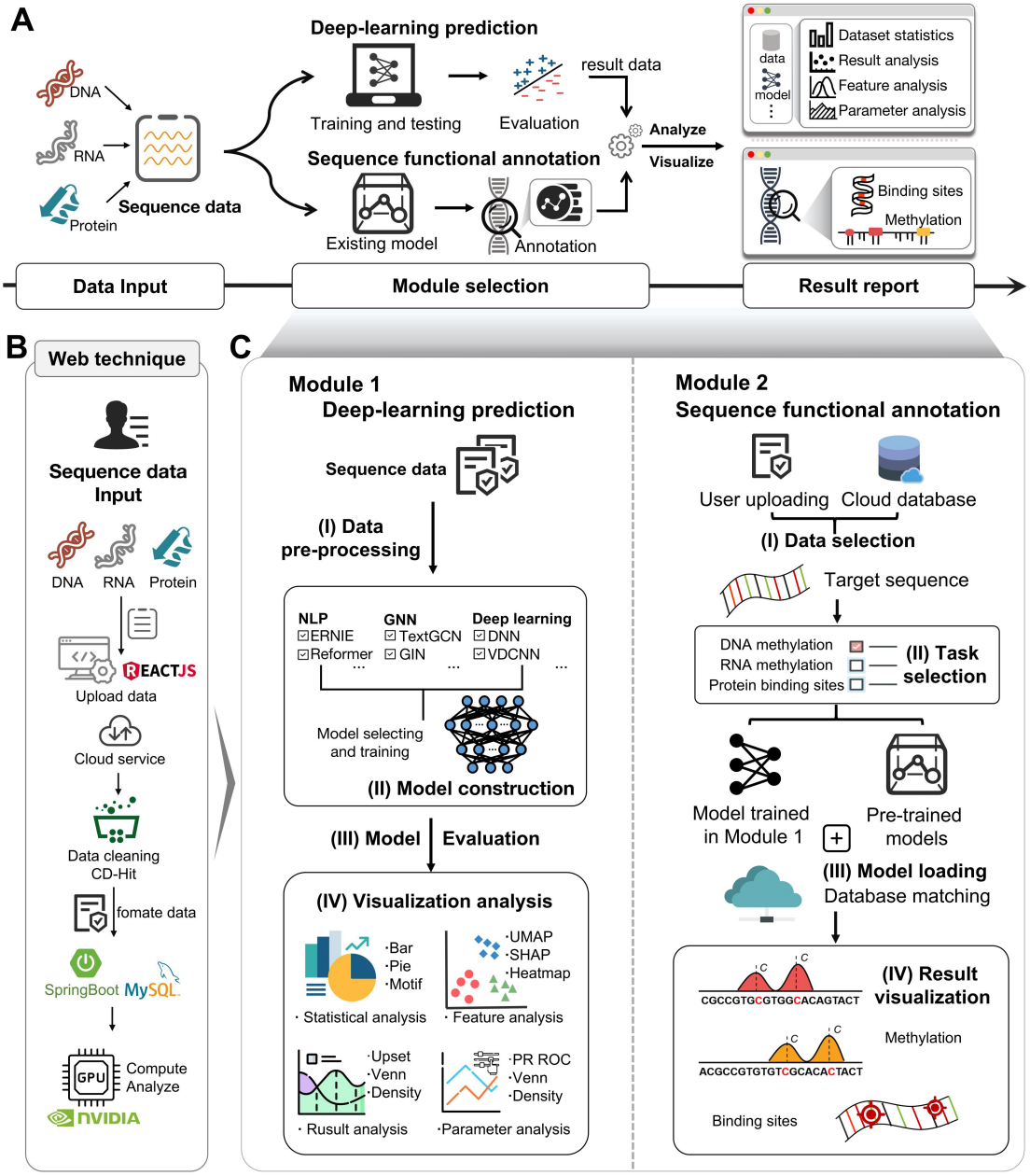


Figure 1. **The organization of the DeepBIO platform**. **(A)** The architecture of the DeepBIO platform. DeepBIO contains four modules, including the data input module, the deep-learning prediction module, the sequence functional annotation module, and the result report module. DeepBIO accepts DNA, RNA, and protein sequence data from the data input module, with some additional user-selectable settings (*e.g.*, sequence similarity setting, imbalanced data processing, and data augmentation). The deep-learning prediction module provides the deep-learning biological sequence binary classification service while the sequence functional annotation module supports the prediction of functional sites for user-provided sequences. In the result report module, DeepBIO provides user-friendly, interactive, and interpretable visualization representations to deliver results of the above two functional modules as a

download-free report. **(B)** The web technique of DeepBIO and an overview of the framework. **(C)** The workflow of two main functional modules. The deep-learning prediction module consists of four steps: (I) Data pre-processing, (II) Model construction, (III) Model evaluation, and (IV) Visualization analysis. DeepBIO outputs classification predictions of input sequences based on the four steps and displays the result reports with interactive graphs and tables; The sequence functional annotation module includes the following steps: (I) Data selection, (II) Task selection, (III) Model loading, and (IV) Result visualization. DeepBIO offers multiple methylation status annotations for DNA and RNA sequences while supporting ligand-binding site recognition for protein sequences.

Data input module

DeepBIO serves as an automatically integrated analysis platform based on biological sequence data, including DNA, RNA, and protein, which can be entered in the online input box or uploaded with the standard file format. Besides, users can select deep-learning models and different settings (*e.g.*, whether to turn on the data enhancement option). After obtaining the user's input data, DeepBIO further performs data cleaning to ensure that the user input data is ready and legal to pass into the next module and run properly.

Deep-learning prediction module

Step 1. Data pre-processing. This step is an optimization-based data processing phase that uses a variety of approaches to handle and obtain the input sequence data that is suitable for model training, ensuring the robustness of the final results and improving the success rate of our service. We design and provide users with four sections to pre-process the input data: (I) Sequence legitimacy testing, (II) Sequence similarity setting, (III) Imbalanced data processing, and (IV) Data augmentation. In Section (I), for the input sequence data uploaded by users, we first examine whether the input data has the legal data format (*e.g.*, FASTA), and then check if the input sequences contain illegal characters (*e.g.*, other characters except for A, T, C, G in DNA sequences). In Section (II), considering high sequence similarity in the input data might lead to performance evaluation bias, we thus provide a commonly used tool named CD-HIT [12] to reduce the sequence similarity in the input data. The similarity threshold ranges from 0.7 to 1. In Section (III), considering most datasets in real scenarios might exist the data imbalance issue (i.e., the positives and negatives are extremely imbalanced), we further provide several imbalanced data processing strategies that are commonly used in the machine learning field, including Focal loss [13], Synthetic Minority Oversampling Technique (SMOTE) [14], and Adaptive Synthetic (ADASYN) [15]. By doing so, we help to prevent the biased prediction of models due to the unbalanced data distribution and significantly improve the comprehensive model performance. Finally, in Section (IV), to alleviate the overfitting problem, which commonly exists in the deep-learning models, we provide some data augmentation methods, such as sequence replacement, sequence flipping, and sequence cropping, which can improve the robustness and generalization ability of the models.

Step 2. Model construction. The pre-processed sequence data is then fed into user-selected models to train and make the binary prediction. **Table 1** summarizes all the state-of-the-art deep-learning models offered on our platform. There are a total of 42 deep learning models,

covering classic deep-learning models (*e.g.*, Deep Neural Networks (DNN), and Gate Recurrent Unit (GRU), *etc*), Natural Language Processing (NLP) models (*e.g.*, Transformer, and Bidirectional Encoder Representations from Transformers (BERT)), and Graph Neural Network (GNN) models (*e.g.*, Graph Convolutional Network (GCN)). Notably, each of the above models can be exploited to deal with all three types (DNA, RNA, and protein) of sequence analysis tasks. In addition, DeepBIO permits users to select several deep-learning models, train and compare their performance in the same data. It's worth noting that we have also deployed several deep-learning models (*e.g.*, DNABERT, RNABERT, and ProtBERT) that are well pre-trained in million-scale datasets. They can be utilized for fine-tuning the models on small datasets, if necessary, to alleviate the overfitting problem and increase the models' generalization capabilities.

Table 1. Deep-learning models provided by DeepBIO

Category	Methods	Refs.
Classic deep-learning methods	DNN	[16]
	RNN	[17]
	LSTM	[18]
	BiLSTM	[19]
	LSTM_Attention	[20]
	GRU	[21]
	TextCNN	[22]
	TextRCNN	[23]
	VDCNN	[24]
	RNN-CNN	[25]
Natural language processing methods	Transformer	[26]
	Reformer	[27]
	Performer	[28]
	Linformer	[29]
	RoutingTransformer	[30]
	DNABERT, RNABERT, ProtBERT	[2, 31]
	BERT-Base	[32]
	BERT-CNN	[33]
	BERT-DPCNN	[34]
	BERT-RNN	
	ERNIE	[35]
Graph neural network methods	GCN	[36]
	TextGNN	[37]
	GIN	[38]
	GAT	[39]
	GraphSage	[40]
	ChebGCN	[41]
	RECT-L	[42]

LightGCN	[43]
GNN-FILM	[44]
HYPER-Conv	[45]
HYPER-Attention	[46]
APPNP	[47]
SGC	[48]
TextRGNN	[49]
TextSCG	[50]
BertGCN	[51]
BertGAT	[52]
RoberTaGCN	[53]
RoBERTaGAT	[54]

Step 3. Model evaluation. To evaluate the prediction performance of models, DeepBIO offers two evaluation methods, including N-fold cross-validation and independent testing. The cross-validation method is used for model evaluation, while the independent testing is used for validating the generalization of the model. Moreover, we choose several commonly used evaluation measures for performance comparison, including Sensitivity (SN), Specificity (SP), Accuracy (ACC), Matthews Correlation Coefficient (MCC), Area Under Receiver Operating Characteristic curve (AUROC), and Area Under Precision Recall Curve (AUPRC). The formulas of these metrics are described as follows:

$$SN = \frac{TP}{TP+FN} \quad \#(1)$$

$$SP = \frac{TN}{TN+FN} \#(2)$$

$$ACC = \frac{TP+TN}{TP+TN+FP+FN} \#(3)$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(FP+FP)(TP+FN)(TN+FP)(TN+FN)}} \quad \#(4)$$

where TP, FP, TN, and FN represent the numbers of true positives, false positives, true negatives, and false negatives, respectively. The AUROC and AUCPRC values, which range from 0 to 1, are calculated based on the ROC curve and the PRC, respectively. The ROC curve shows the proportion of true positives versus false positives, in which the area under the ROC curve (AUROC) equals the probability of ranking a randomly chosen true target higher than a randomly chosen decoy target. In comparison to the ROC curve, the PRC is used more frequently to evaluate the performance of a model on imbalanced datasets. Similarly, we can calculate the area under the PRC (AUPRC), which equals the average of the interpolated precisions. The higher the AUROC and AUPRC values, the better the predictive performance of the underlying model.

Step 4. Visualization analysis. DeepBIO combines the prediction results and intermediate data analysis generated from above and helps users to better understand the input data through the statistical analysis and the model results through various types of visualization analyses. To provide users with intuitive and comprehensive result analysis, we design and illustrate multiple visual presentation formats, including the pie plot, histogram, sequence logo graph, ROC and PR curves, kernel density plot, and scatter plot. In Table 2, we display the four sections of the visualization analysis, which are statistical analysis, result analysis, feature analysis, and parameter optimization analysis. The details of each section are described as follows.

In statistical analysis, we extract and evaluate input datasets themselves to assist users in better comprehending the statistical distribution of training and testing datasets. In particular, we display three types of figures: sequence composition analysis, sequence motif analysis, and the distribution of different sequence lengths. In sequence composition analysis, DeepBIO plots the sequence composition histogram by counting the nucleic acid in DNA or RNA sequences and the amino acid in proteins of the incoming positive and negative dataset examples. Besides, if the sequence length of input data is not identical, DeepBIO plots both the motif of the sequence statistics and the distribution of different sequence lengths, otherwise only the motif of the sequence statistics.

The DeepBIO visualization analysis includes plots considering all the evaluation metrics presented in step 3, to conduct detailed performance comparisons among different models, such as ROC and PR curves, the performance of different epochs, density distribution of model prediction confidence, *etc*. Besides, we offer a Venn diagram and an Upset plot to illustrate the relationships among different models from the prediction results. It can be concluded that in this part users can get an intuitive comparison of prediction performance for different deep-learning models.

For feature analysis, we use a dimensionality reduction tool Uniform Manifold Approximation and Projection (UMAP) [55] for feature space valuation analysis, and a commonly used tool Shapley Additive explanation (SHAP) [56] for feature importance analysis. UMAP can map and visualize high-dimensional features to a low-dimensional space while better retaining the global structure with a superior runtime performance, and better scalability than other similar methods. On the other hand, SHAP gives a good explanation of the importance of different features extracted by the model and the extent of their impact on the dataset. Besides, for some specific models like DNABERT and RNABERT, users can explore what the model has learned from the dataset by visualizing the attention heatmap of the user-choosing biological sequence. Note that the element in the heatmap represents the correlation degree of two positions along the sequences. Furthermore, we provide users with conservative sequence motifs, which are important for DNA or RNA methylation prediction from the dataset. The process of discovering conservative sequence motifs is described as follows. Firstly, we apply the attention mechanism of BERT-based models to identify key regions from the dataset. Then, for those identified regions, we further extracted and visualized the corresponding motifs using the attention scores obtained from the model. In this way, we highlight the concerns of different

models for different features and enhance the interpretability of the deep-learning models.

For parameter optimization analysis, we provide users with some parameter options to make their own choice to adjust the model training process, including the effect of different sequence similarities in datasets on predictive performance, the effect of different data augmentation strategies on predictive performance, and the effect of different imbalanced data processing methods on predictive performance. Besides, we also give some tabular information, including a summary of the input datasets and the performance of deep-learning models. This helps the users to compare the results of the same model with different parameters.

Table 2. Result visualization analysis in the prediction module

Category	Type	Purpose
Statistical analysis	Pie plot	Sequence compositional analysis
	Sequence statistics	Sequence statistical analysis from the datasets
	Histogram	Distribution of different sequence length
Result analysis	ROC and PRC curve	Performance of model prediction classification
	Line chart	Performance of different epochs of models
	Kernel density plot	Predict confidence by different models
	Venn diagram and Upset plot	Overlap of prediction results of different models
Feature analysis	ROC and PRC curve	Feature performance comparison between hand-crafted features and the feature learned by our models
	UMAP plot	Feature analysis result by UMAP (Uniform Manifold Approximation and Projection) [55]
	SHAP plot	Feature importance analysis by SHAP (Shapley Additive explanation)[56]
	Heatmap	Feature representation attention analysis
	Motifs	The discovery of conservative sequence motifs by models
Parameter Optimization analysis	ROC and PRC curve, Histogram	The effect of different parameters (similarity in datasets, data augmentation strategies, imbalanced data processing methods) on predictive performance

Sequence functional annotation module

To extend our capability of functional site discovery and pave the way to formulate the relationship between sequence and function, we creatively added a sequence functional annotation module to our DeepBIO platform. Parallel to the deep-learning prediction module,

the sequence functional annotation module has four steps and can be adopted in the methylation status annotation for DNA and RNA sequences or the ligand-binding site recognition for protein sequences. The details of this module are as follows.

Step 1. Data selection. In this step, DeepBIO needs users to provide and upload the sequences they want to annotate. Particularly, for the DNA methylation prediction task, we support two ways to input sequence data, that is: (1) uploading sequence data by themselves, (2) and selecting a segment of our preset human gene by choosing a specific cell line and chromosome.

Step 2. Task selection. There three types of functional annotation tasks: DNA methylation, RNA methylation, and protein-ligand binding. For DNA methylation, there are DNA N4-methylcytosine (4mC) prediction, DNA 5-methylcytosine (5mc) prediction, and DNA N6-methyladenine (6mA) prediction. For RNA methylation, there are RNA 5-hydroxymethylcytosine (5hmC) prediction, RNA N6-methyladenosine (m6A) prediction, and RNA N7-methylguanosine (m7G) prediction. For protein-ligand binding, there are DNA-binding prediction, RNA-binding prediction, and Peptide-binding prediction.

Step3. Model loading. In this step, users can select our default models or their models trained in previous tasks. For the default models, we provide several deep-learning pre-trained models, such as DNABERT. For the user-trained models, they can choose one model in previously completed tasks. It is noted that if users select models by themselves, the type of analyzed biological sequences and task must be the same as that in the previously completed task.

Step 4: Result visualization. In this part, we performed result visualization analysis and provided an in-depth comparison study between the predictions by our models and experimental data to lessen the user's burden of analysis. Specifically, for DNA, we designed two visualization sections, including functional site annotation and integrative analysis. For functional site annotation, we predict the functional sites that match the selected task type (*e.g.*, DNA 4mC prediction) and show the model predicted confidence of the corresponding position. In integrative analysis, we perform statistical analysis of the expression of DNA methylation and different histone modification signals. For RNA, due to the constraint of data, we only perform the functional site annotation section same as in DNA. In addition, for protein, to intuitively show the binding site annotation results, we visualize and annotate the binding residues on the 3D structure of a given protein sequence with PDB ID.

Result report module

We list the analysis results in the form of reports, which involve various kinds of data formats from the results of the two main functional modules. The users can view or download freely all of the result data and charts. Notably, each submitted task is immediately notified to the corresponding user upon completion, and the results are kept for two months.

CASE STUDY

We showcase real-world applications of DeepBIO with two bioinformatic scenarios: identification of the m6A methylation and prediction of the protein-peptide interaction. We emphasize that the underlying objective is to illustrate how to use our platform for two such diverse applications, rather than securing the top predictive performance compared to the state-of-the-art.

DNA methylation is essential for normal development and is closely associated with a variety of key processes, ranging from genomic imprinting, and aging, to carcinogenesis. Therefore, for the task of the m6A methylation prediction, we use the Rosa chinensis dataset with 600 methylation sequences and 600 non-methylation sequences as an example dataset of DNA. In addition, the prediction of protein-peptide binding residues and interactions can provide significant promotion in new drug design and cancer research. We choose a dataset with 300 peptide-binding sequences and 300 non-peptide-binding sequences as a protein example dataset. With the two example datasets, we first randomly split them into the training set and testing set and then input them into our online platform to demonstrate the data analysis and integration functions of DeepBIO. More details of the visualization results of the two cases are described below.

Dataset statistics

To help the users better understand the training and testing datasets they input, we especially retrieve and analyze the datasets. Specifically, we start by separating the positive and negative samples from the training and testing sets, respectively. The proportion and number of bases or amino acids within them are then counted and illustrated as a pie plot and composition histogram (**Figure 2A**). In addition, we also calculate and show the sequence length distribution (**Figure 2B**), from which we can learn how the sequence length influences positive and negative samples. Furthermore, we use the Weblogo tool to generate the motifs of sequence statistics from the datasets to analyze the composition at the sequence level (**Figure 2C**).

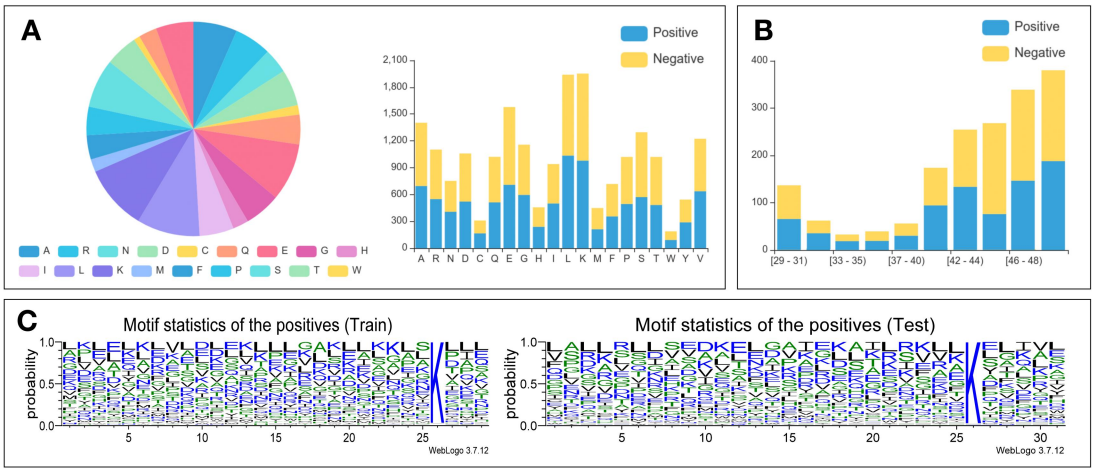
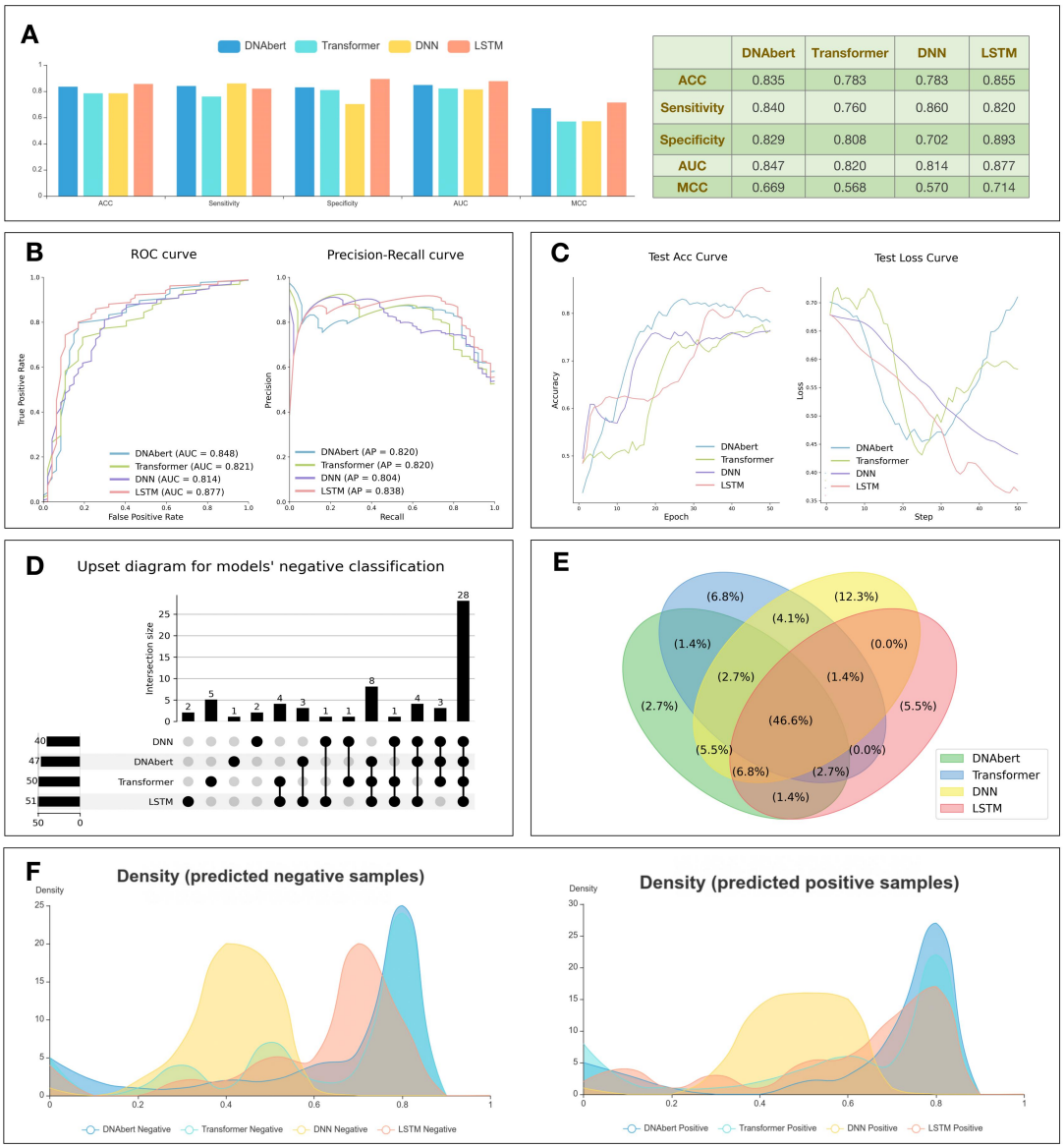


Figure 2. The visualization generated by using DeepBIO to apply statistical analysis to datasets. (A) Sequence compositions of positive and negative samples in the training and testing datasets; **(B)** Sequence length distribution of positive and negative samples in the training and testing datasets; **(C)** Motifs of sequence statistics from the training and testing

datasets.

Model prediction analysis

DeepBIO provides detailed analysis for users to interpret and show comprehensive performance comparisons among different deep-learning models. Here, we select four models to compare their prediction performance on our example dataset. **Figure 3A** shows basic comparison results, which include commonly used binary classification indicators, such as ACC, Sensitivity, Specificity, AUC, and MCC. It can be seen that LSTM achieves the best ACC value of 0.855, outperforming other models by over 5%. To ease the evaluation of models, we integrate the receiver operating characteristic (ROC) plot and precision-recall (PR) plot (**Figure 3B**), to illustrate the prediction performance of different models from a comprehensive perspective. Here, the LSTM has a better performance than other models again which agrees with the conclusion in **Figure 3A**. To learn more about the trend of model performances in the training process, we draw the epoch plot (**Figure 3C**) to show how the accuracy and calculation loss change in the whole training process of each model. From **Figure 3C**, it is easy to find that the accuracy of each model increases rapidly at first, and then flat out gradually due to overfitting, which can be illustrated by the trend of calculation loss. Also, we can see the more parameters the model has like DNABERT and Transformer, the faster the overfitting speed they will own. In addition, we further provide an Upset plot and Venn diagram (**Figure 3D-E**) to illustrate the relationships among different models from the prediction results. Moreover, the density distribution of the prediction confidence by different deep-learning models (**Figure 3F**) will bring users a clear understanding of the predicting preference of each model. For example, DNN focuses its preference on 0.5 which is the center part of the X label, meaning that its poor binary classification ability and its confusion on the difference between positive and negative.



	DNAbert	Transformer	DNN	LSTM
ACC	0.835	0.783	0.783	0.855
Sensitivity	0.840	0.760	0.860	0.820
Specificity	0.829	0.808	0.702	0.893
AUC	0.847	0.820	0.814	0.877
MCC	0.669	0.568	0.570	0.714

Figure 3. Using DeepBIO to train selected models and obtain predictions for results analysis. (A) Performance comparison among different models in terms of ACC, Sensitivity, Specificity, AUC, and MCC; **(B)** ROC and PR curves of different deep-learning models; **(C)** The trends of performance and calculation loss with each epoch, showing ACC and loss change process on different models; **(D-E)** Upset plot and Venn diagram to express the relationships of prediction results for different models; **(F)** Density distribution of the prediction confidence for different deep-learning models.

Feature analysis and visualization

To compare the features automatically learned by deep-learning models in our end-to-end training process and the traditional hand-crafted features, we select ANF [57], binary [58], CKSNAP [59], and DNC [60, 61] approaches for nucleotide sequences and BLOSUM62 [62], CKSAAGP [59, 63], and CTDC [64] approaches for protein sequences among a variety of hand-crafted features. **Figure 4A-B** show the prediction performance in terms of many metrics

such as ACC, Sensitivity, Specificity, AUC, and MCC, allowing us to easily conclude that the features derived from the deep-learning methods generally are superior to hand-crafted features. In addition, to intuitively visualize and evaluate the feature representations extracted from different models, we use the UMAP plot, which provides superior dimension reduction while preserving more of the global structure (**Figure 4C**). Each point in the diagram represents a sample in the dataset, and the positive and negative classes are distinguished by different colors. In this case, we can get a better sense of the classification ability of each trained-over model by looking at this feature space visualization plot. Moreover, we also provide the SHAP plot (**Figure 4D**), trying to interpret the importance of different features extracted by the model. Each row represents a dimension of the feature representation, and the abscissa refers to the SHAP value. Higher values of the feature are represented with redder colors, and lower values of the feature are bluer colors. Therefore, the SHAP plot shows the relationship between the value of features and their impact on the model prediction. Furthermore, in **Figure 4E**, we calculate the importance of the base at each position on the user-given sequence from the attention matrix in the model and visualize it with normalized attention scores, enabling us to analyze which region the model considers more important for the prediction. To improve the interpretability in our analysis and further study what the models have learned from the training, we extract the important regions using the model confidence probability through the attention mechanism to find conservative sequence motifs **(Figure 4F**) from training datasets, which are widely assumed to be related to biological functions and can further be used to match the corresponding functional patterns in motif databases.

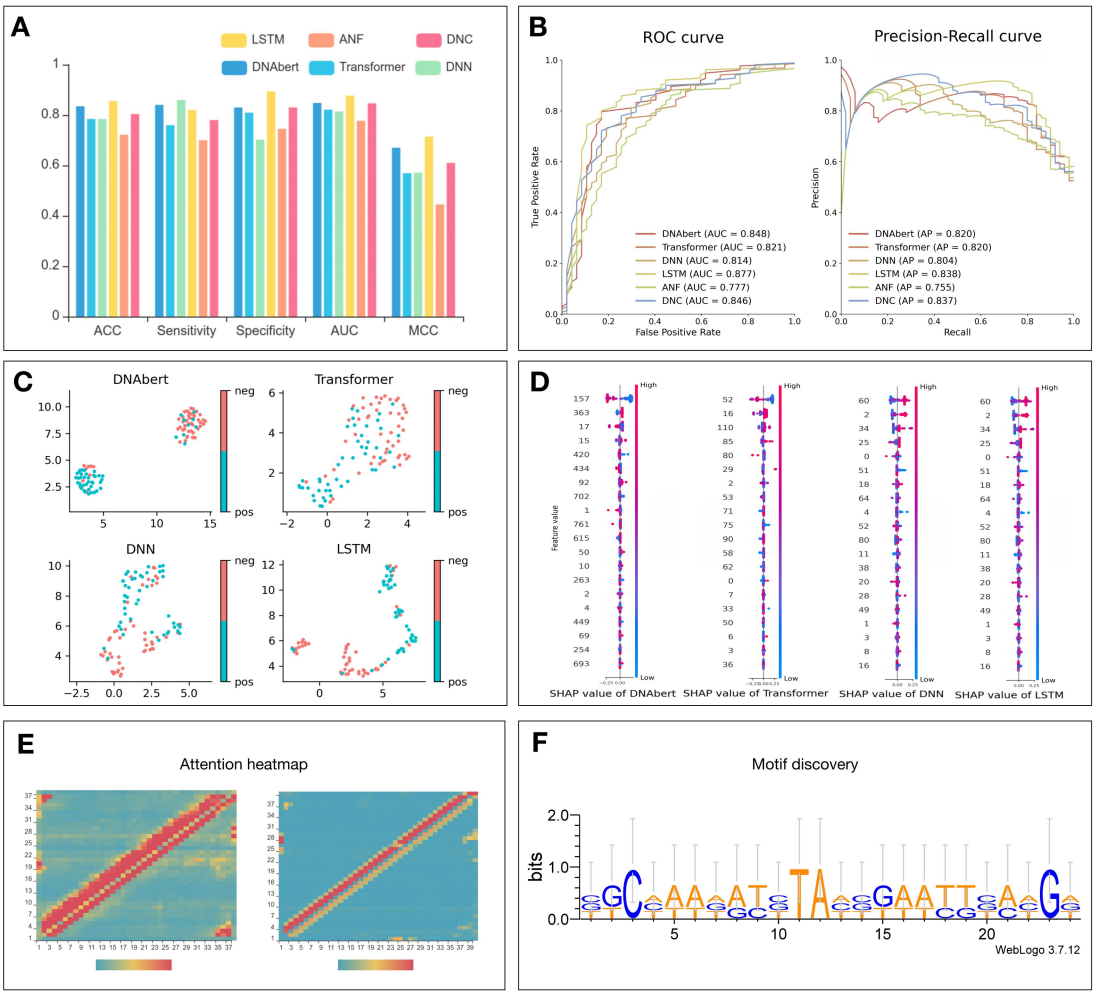


Figure 4. Using DeepBIO to train selected models and to obtain visualization applied to feature analysis. (A) Prediction performance comparison between deep-learning models and traditional hand-crafted feature-based methods; **(B)** ROC and PR curves of deep-learning models and traditional hand-crafted feature-based methods; **(C)** UMAP feature visualization results of different deep-learning models; **(D)** SHAP feature importance analysis of different deep-learning models; **(E)** Heatmap plot of the user-choosing biological sequence; **(F)** Motif discovery by deep-learning models.

Functional annotation analysis

This part is designed for functional annotation of biological sequences, including functional site prediction and result visualization analysis. To improve the versatility of our platform, we provide users two ways to annotate, that is, using the previously trained models in their former submitted tasks or our default models. In addition, for the input data of DNA sequence annotation tasks, users have two options: one is to select a gene range from our preset human genes by choosing a specific cell line and chromosome, and the other is to submit their sequences. Afterward, we can predict DNA methylation sites based on the input and display their confidence and location obtained from the models, as shown in **Figure 5A**. Furthermore, if users select a segment from our preset human gene, we will display the comparison between our predictions and relevant histone modification signals queried from the database (**Figure 5B**), assisting in mining the potential relations between them. Moreover, for protein

sequence annotation tasks, we predict and visualize the binding site annotation results on the 3D structure of the given protein sequence with PDB ID 1a81A (**Figure 5C**).

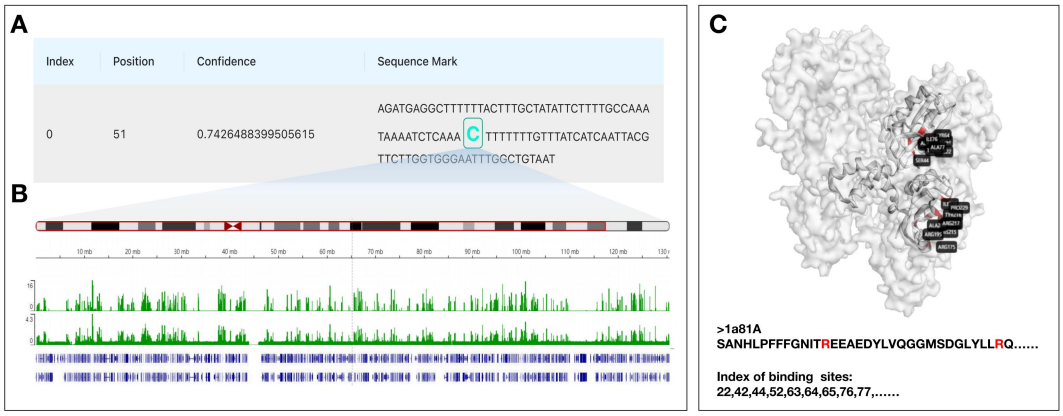


Figure 5. A visualization generated using DeepBIO to obtain analysis for functional annotation. (A) Prediction results of DNA methylation annotation tasks; **(B)** Comparison and analysis between the DNA methylation site prediction results and relevant histone modification signals; **(C)** Prediction results of protein-binding site annotation tasks.

Webserver Implementation

DeepBIO is empowered by the professional high-performance graphic processing architecture with million-scale calculating capability. Specifically, DeepBIO is deployed with Ubuntu 18.04 Linux system, multi Intel Xeon Silver 4210R CPUs, 256GB RAM, hundreds of TB Solid State Drive and NVIDIA Tesla V100 GPU clusters. Unlike other online platforms that use CPUs for training and predicting, the models we provide are all GPU-based, which significantly reduces the time used for task training. It can be seen from **Supplementary Table 1** that we can now spend less time optimizing deep-learning models and enable fast and accurate predictions. In addition, **Supplementary Figures 1-6** show that we have carefully designed the front-end page to give users an interactive visualization of the results while they can download each result graph freely. To be specific, we use React framework to set up the user interface, Spring Boot framework for server implementation in the back end, python and R programming language for the model's construction and visualization, and MySQL database to manage the data storage. Besides, interactive expression graphs were constructed to visualize the analysis result on the front page. Our platform can run smoothly on all major browsers including Internet Explorer (≥v.7.0), Mozilla Firefox, Microsoft Edge, Safari, and Google Chrome. Moreover, we support concurrent access while the back end can also achieve multiple tasks at the same time, avoiding performance problems caused by the simultaneous submission of tasks. In conclusion, with our DeepBIO platform, data scientists and researchers limited by equipment resources can now afford high-performance computers to tackle their most challenging work.

AVAILABILITY OF DATA AND MATERIALS

As an online platform for biological sequence analysis, DeepBIO can be freely accessed without registration via http://inner.wei-group.net/DeepBIO. All code used in data analysis and preparation of the manuscript, alongside a description of necessary steps for reproducing results, can be found in a GitHub repository accompanying this manuscript: https://github.com/WLYLab/DeepBIO.

Competing interests

The authors declare that they have no competing interests.

Author contributions

L.W. conceived the basic idea and de-signed the DeepBIO framework. R.W., J.J., C.Y., H.Y., F.W., J.F. set up the server and designed visualization of result analysis. R.W., Y.J., L.W., C.Y. designed the interface and tested the server. R.W, C.Y., J.J., L.W. wrote the manuscript.

Funding

The work was supported by the Natural Science Foundation of China (Nos. 62072329, and 62071278).

Reference

1. Larranaga, P., et al., *Machine learning in bioinformatics.* Brief Bioinform, 2006. **7**(1): p. 86-112.
2. Elnaggar, A., et al., *ProtTrans: towards cracking the language of Life's code through self-supervised deep learning and high performance computing.* 2020.
3. Wang, R., et al., *Predicting protein-peptide binding residues via interpretable deep learning.* Bioinformatics, 2022. **1**: p. 10.
4. Chen, Z., et al., *iLearnPlus: a comprehensive and automated machine-learning platform for nucleic acid and protein sequence analysis, prediction and visualization.* Nucleic Acids Res, 2021. **49**(10): p. e60.
5. Cao, D.-S., et al., *Rcpi: R/Bioconductor package to generate various descriptors of proteins, compounds and their interactions.* Bioinformatics, 2015. **31**(2): p. 279-281.
6. Xiao, N., et al., *protr/ProtrWeb: R package and web server for generating various numerical representation schemes of protein sequences.* Bioinformatics, 2015. **31**(11): p. 1857-1859.
7. Avsec, Ž., et al., *The Kipoi repository accelerates community exchange and reuse of predictive models for genomics.* Nature biotechnology, 2019. **37**(6): p. 592-600.
8. Budach, S. and A. Marsico, *Pysster: classification of biological sequences by learning sequence and structure motifs with convolutional neural networks.* Bioinformatics, 2018. **34**(17): p. 3035-3037.
9. Chen, K.M., et al., *Selene: a PyTorch-based deep learning library for sequence data.* Nature methods, 2019. **16**(4): p. 315-318.
10. Li, C., et al., *A novel computational model for predicting microRNA–disease associations*

based on heterogeneous graph convolutional networks. Cells, 2019. **8**(9): p. 977.

11. Ji, Y., et al., *DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome.* Bioinformatics, 2021. **37**(15): p. 2112-2120.
12. Li, W. and A. Godzik, *Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences.* Bioinformatics, 2006. **22**(13): p. 1658-9.
13. Lin, T.-Y., et al. *Focal loss for dense object detection*. in *Proceedings of the IEEE international conference on computer vision*. 2017.
14. Chawla, N.V., et al., *SMOTE: synthetic minority over-sampling technique*. Journal of artificial intelligence research, 2002. **16**: p. 321-357.
15. He, H., et al. *ADASYN: Adaptive synthetic sampling approach for imbalanced learning*. in *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*. 2008. IEEE.
16. LeCun, Y., Y. Bengio, and G. Hinton, *Deep learning.* nature, 2015. **521**(7553): p. 436-444.
17. Rumelhart, D.E., G.E. Hinton, and R.J. Williams, *Learning representations by back-propagating errors.* nature, 1986. **323**(6088): p. 533-536.
18. Hochreiter, S. and J. Schmidhuber, *Long short-term memory.* Neural computation, 1997. **9**(8): p. 1735-1780.
19. Graves, A. and J. Schmidhuber, *Framewise phoneme classification with bidirectional LSTM and other neural network architectures.* Neural networks, 2005. **18**(5-6): p. 602-610.
20. Wang, Q. and Y. Hao, *ALSTM: An attention-based long short-term memory framework for knowledge base reasoning.* Neurocomputing, 2020. **399**: p. 342-351.
21. Dey, R. and F.M. Salem. *Gate-variants of gated recurrent unit (GRU) neural networks*. in *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*. 2017. IEEE.
22. Dos Santos, C. and M. Gatti. *Deep convolutional neural networks for sentiment analysis of short texts*. in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. 2014.
23. Lai, S., et al. *Recurrent convolutional neural networks for text classification*. in *Twenty-ninth AAAI conference on artificial intelligence*. 2015.
24. Simonyan, K. and A.J.a.p.a. Zisserman, *Very deep convolutional networks for large-scale image recognition.* 2014.
25. Wang, J., et al. *Cnn-rnn: A unified framework for multi-label image classification*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
26. Vaswani, A., et al. *Attention is all you need*. in *Advances in neural information processing systems*. 2017.
27. Kitaev, N., Ł. Kaiser, and A.J.a.p.a. Levskaya, *Reformer: The efficient transformer.* 2020.
28. Choromanski, K., et al., *Rethinking attention with performers.* 2020.
29. Wang, S., et al., *Linformer: Self-attention with linear complexity.* 2020.
30. Roy, A., et al., *Efficient content-based sparse attention with routing transformers.* Transactions of the Association for Computational Linguistics, 2021. **9**: p. 53-68.
31. Ji, Y., et al., *DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome.* Bioinformatics, 2021.
32. Devlin, J., et al., *Bert: Pre-training of deep bidirectional transformers for language understanding.* arXiv preprint arXiv:1810.04805, 2018.
33. Safaya, A., M. Abdullatif, and D. Yuret. *Kuisail at semeval-2020 task 12: Bert-cnn for offensive*

speech identification in social media. in *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. 2020.

34. Li, Y.-J., et al., *Microblog Rumor Detection Based on Bert-DPCNN*, in *Artificial Intelligence in China*. 2021, Springer. p. 524-530.
35. Zhang, Z., et al., *ERNIE: Enhanced language representation with informative entities.* arXiv preprint arXiv:1905.07129, 2019.
36. Kipf, T.N. and M.J.a.p.a. Welling, *Semi-supervised classification with graph convolutional networks.* 2016.
37. Zhu, J., et al. *Textgnn: Improving text encoder via graph neural network in sponsored search*. in *Proceedings of the Web Conference 2021*. 2021.
38. Chen, J., et al. *Generative invertible networks (GIN): Pathophysiology-interpretable feature mapping and virtual patient generation*. in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. 2018. Springer.
39. Wang, K., et al., *Relational graph attention network for aspect-based sentiment analysis.* arXiv preprint arXiv:2004.12362, 2020.
40. Hamilton, W., Z. Ying, and J. Leskovec, *Inductive representation learning on large graphs.* Advances in neural information processing systems, 2017. **30**.
41. Defferrard, M., X. Bresson, and P. Vandergheynst, *Convolutional neural networks on graphs with fast localized spectral filtering.* Advances in neural information processing systems, 2016. **29**.
42. Wang, Z., et al., *Network embedding with completely-imbalanced labels.* IEEE Transactions on Knowledge and Data Engineering, 2020. **33**(11): p. 3634-3647.
43. He, X., et al. *Lightgcn: Simplifying and powering graph convolution network for recommendation*. in *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 2020.
44. Brockschmidt, M. *Gnn-film: Graph neural networks with feature-wise linear modulation*. in *International Conference on Machine Learning*. 2020. PMLR.
45. Ma, T., A.V. Dalca, and M.R. Sabuncu. *Hyper-convolution networks for biomedical image segmentation*. in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2022.
46. Duan, W., et al., *Injecting descriptive meta-information into pre-trained language models with hypernetworks.* 2021.
47. Klicpera, J., A. Bojchevski, and S. Günnemann, *Predict then propagate: Graph neural networks meet personalized pagerank.* arXiv preprint arXiv:1810.05997, 2018.
48. Pho, P. and A.V. Mantzaris, *Regularized Simple Graph Convolution (SGC) for improved interpretability of large datasets.* Journal of Big Data, 2020. **7**(1): p. 1-17.
49. Chen, J., et al., *TextRGNN: Residual Graph Neural Networks for Text Classification.* arXiv preprint arXiv:2112.15060, 2021.
50. Wu, F., et al. *Simplifying graph convolutional networks*. in *International conference on machine learning*. 2019. PMLR.
51. Lin, Y., et al., *BertGCN: Transductive Text Classification by Combining GCN and BERT.* 2021.
52. Veličković, P., et al., *Graph attention networks.* 2017.
53. Wei, M., Y. He, and Q. Zhang. *Robust layout-aware IE for visually rich documents with pre-trained language models*. in *Proceedings of the 43rd International ACM SIGIR Conference*

on Research and Development in Information Retrieval. 2020.

54. Chandra, S., et al., *Graph-based modeling of online communities for fake news detection.* arXiv, 2020.

55. Becht, E., et al., *Dimensionality reduction for visualizing single-cell data using UMAP.* Nature biotechnology, 2019. **37**(1): p. 38-44.

56. Lundberg, S.M. and S.-I. Lee. *A unified approach to interpreting model predictions*. in *Proceedings of the 31st international conference on neural information processing systems*. 2017.

57. Chen, W., et al., *Identification and analysis of the N6-methyladenosine in the Saccharomyces cerevisiae transcriptome.* Scientific reports, 2015. **5**(1): p. 1-8.

58. Chen, Z., et al., *Prediction of ubiquitination sites by using the composition of k-spaced amino acid pairs.* PloS one, 2011. **6**(7): p. e22930.

59. Chen, Z., et al., *iLearn: an integrated platform and meta-learner for feature engineering, machine-learning analysis and modeling of DNA, RNA and protein sequence data.* Briefings in bioinformatics, 2020. **21**(3): p. 1047-1057.

60. Liu, B., et al., *Pse-in-One: a web server for generating various modes of pseudo components of DNA, RNA, and protein sequences.* Nucleic acids research, 2015. **43**(W1): p. W65-W71.

61. Liu, B., et al., *repDNA: a Python package to generate various modes of feature vectors for DNA sequences by incorporating user-defined physicochemical properties and sequence-order effects.* Bioinformatics, 2015. **31**(8): p. 1307-1309.

62. Lee, T.-Y., et al., *Incorporating distant sequence features and radial basis function networks to identify ubiquitin conjugation sites.* PloS one, 2011. **6**(3): p. e17331.

63. Zhou, C., et al., *Identification and analysis of adenine N6-methylation sites in the rice genome.* Nature plants, 2018. **4**(8): p. 554-563.

64. Cai, C., et al., *SVM-Prot: web-based support vector machine software for functional classification of a protein from its primary sequence.* Nucleic acids research, 2003. **31**(13): p. 3692-3697.