BioSeq-BLM: a platform for analyzing DNA, RNA, and protein sequences based on biological language models

Manual of stand-alone tools of BioSeq-BLM

2021-03

Home page: http://bliulab.net/BioSeq-BLM/

Content

1 Introduction

For this section, we will introduce the sequence analysis tool, **BioSeq-BLM**. The **BioSeq-BLM** is a package for DNA, RNA and protein sequence analysis based on biological language models (BLMs). We incorporate 155 different BLMs for DNA, RNA and protein sequence analysis, and extend these BLMs into a system called **BioSeq-BLM**, which is able to automatically represent and analyze the sequence data only requiring the sequence data in FASTA format as inputs. More details will be introduced in the following parts of the manual.

2 Installation

2.1 For Windows

The Windows 7 or later versions are supported.

Before using **BioSeq-BLM**, the Python software should be firstly installed and configured. Python 3.x 64-bit (especially 3.7 64-bit) are recommended for corresponding stand-alone package, which can be download from https://www.python.org.

After installed the Python, the Python package Numpy (1), SciPy (2), and matplotlib (3) should be downloaded from http://www.lfd.uci.edu/~gohlke/pythonlibs/, or use the following command to install:

>pip install numpy

>pip install SciPy

>pip install matplotlib

The Python package scikit-learn (4) should be downloaded and installed from http://scikit-learn.org/dev/install.html, or use the following commands if Internet is accessible:

> pip install scikit-learn

The Python package imbalanced-learn (5) can be installed by using this command line:

>pip install imbalanced-learn

The Pytorch(6) can be installed by using this command line:

>pip install torch

2.2 For Linux

For Linux operating system, the python should be configured as Windows firstly.

If your Linux operating system does not have scikit-learn, numpy, scipy, matplotlib, imbalanced-learn, pandas, networkx, genism and torch, you should use the commods as follows:

>sudo apt-get install scikit-learn
>sudo apt-get install numpy
>sudo apt-get install scipy
>sudo apt-get install matplotlib
>sudo apt-get install imbalanced-learn
>sudo apt-get install torch

2.3 Not Necessary Software

Some external softwares are also needed for generating a few specific features, you can download via following link and put them in the directory "software".

Basic Local Alignment Search Tool: BLAST (7) (https://blast.ncbi.nlm.nih.gov/Blast.cgi?CMD=Web&PAGE_TYPE=BlastDocs&DOC_TYPE=Download).

The predicted secondary structure features are generated by software PSIPRED (8) (9), which can be downloaded from http://bioinfadmin.cs.ucl.ac.uk/downloads/psipred/.

The solvent accessible surface area features is generated by SPIDER2 (10), which can be downloaded from http://sparks-lab.org/pmwiki/download/index.php?Download=yueyang/SPIDER2_local.tgz

The tool ViennaRNA (11) (https://www.tbi.univie.ac.at/RNA/) is needed for calculating RNA second structure features. You can download it from website and add the root directory to the PATH environment variable.

The sequence conservation score features are generated by the package rate4site (12), which can be installed by the following command:

>sudo apt-get install rate4site

Now, BioSeq-BLM is ready to use!

3 Function description

3.1 Directory structure

As shown in **Figure S1**, there are five folders in **BioSeq-BLM** stand-alone package: "code", "docs", "example", "results" and "software".

The "code" directory contains several Python files and folders. The Python scripts in "FeatureExtractionMode" folder used for generating feature vectors based on the input sequence files and the selected feature extraction mode. The Python scripts in "MachineLearningAlgorithm" folder used for constructing classifier or sequence

labelling model. The "FeatureExtractionSeq .py" and "FeatureExtractionRes .py" are used for calling the function in the directory "FeatureExtractionMode" to generate feature vectors for biological sequence at sequence level or residue level. Similarly, "MachineLearningSeq.py" and "MachineLearningRes.py" are used to construct predictor for specific feature vectors. "FeatureAnalysis.py" is used for analyzing feature vectors. "SemanticSimilarity.py" is a script for calculating semantic similarity for training dataset and testing dataset. "BioSeq-BLM_Seq.py" and "BioSeq-BLM_Res.py" are executive Python scripts used for achieving the one-stop function. "CheckAll.py" is used for checking parameters for feature extraction mode and machine learning algorithm and contains the constants used in the scripts.

The "example" folder contains the dataset files used in the example and the "results" folder is used to store the generated model file, feature vectors files and other output files in cross validation process and independent test process. The "docs" folder contains the release note of **BioSeq-BLM**.

The "software" folder contains the external software like FlexCRFs and PSIPRED (if necessary). You can download the external software and configured with the assistance of installation.

Pay attention, the modifications of directory structure are not suggested.

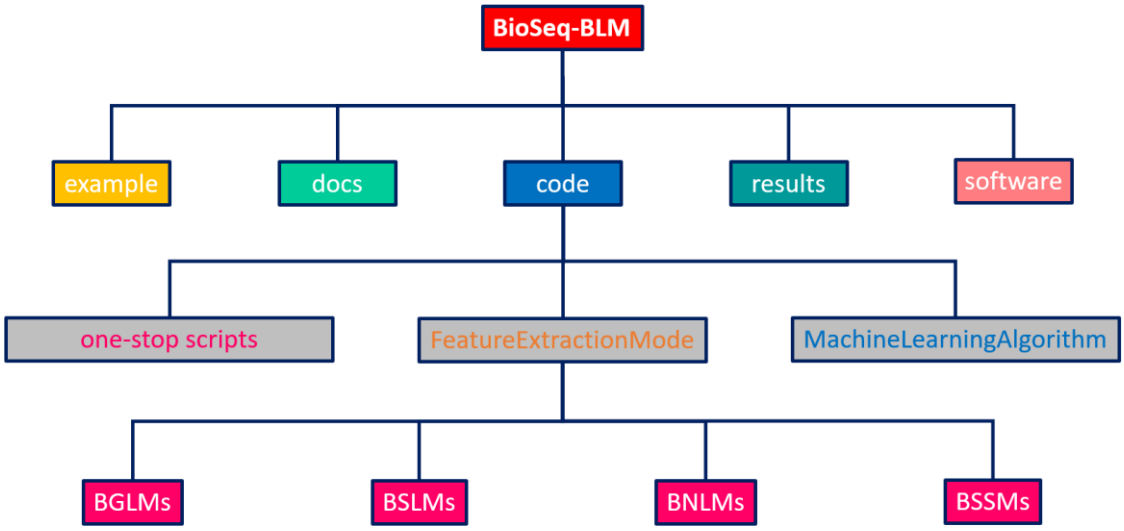

Figure S1. The main modules of the BioSeq-BLM.

3.2 Scripts for feature extraction based on BLMs

3.2.1 Method for feature extraction

For the detailed information of feature extraction based on BLMs, please refer to the description part in web server (http://bliulab.net/BioSeq-BLM/doc/).

3.2.2 Scripts

"**FeatureExtractionRes.py**"and "**FeatureExtractionSeq.py**" are two executive Python scripts used for generating feature vectors based on biological language models at residue level and sequence level.

Input

The input file for "**FeatureExtractionRes.py**" should be a sequence file and a label file. The input file for "**FeatureExtractionSeq.py**" should be sequence file(s). The sequence file should be in a valid FASTA format that consists of a single initial line beginning with a greater-than symbol (">") in the first column, followed by lines of sequence data. The label file should be in a valid FASTA format that consists of a single initial line beginning with a greater-than symbol (">") in the first column, followed by lines of label data. The words right after the ">" symbol in the single initial line are optional and only used for the purpose of sequence identification and description.

For example, the valid FASTA format is as follows:

Sequence Input:

```
>example
gaccagcttttaaaccgactccgtgctactgacgacca
```

Label Input:

```
>example
1 0 0 0 0 1 0 1 1 1 1 0 0 0 0 0 1 0 0 1 0 0 1 1 1 0 1 0 0 1 1 0 0 1 0 0 0 0
```

Additional input

Physicochemical Properties Selection

The Physicochemical Properties Selection file is a text file that contains a list of property names used for generating the modes in categories: autocorrelation, pseudo nucleotide composition/ pseudo amino acid composition. For example, if you want to use the "Rise", "Tilt" and "Shift" of DNA dinucleotide for calculating, the Physicochemical Properties Selection file should be written as follows:

```
Rise
Tilt
Shift
```

After saving this file as "propChosen.txt" and specifying it using the command "-i propChosen.txt", or just "I propChosen.txt", the above three properties will be used in calculations. Meanwhile, you can also use the command "-a True" to select all the built-in physicochemical properties for the corresponding sequence type, which can be selected by using parameter DNA, RNA or PROTEIN.

The complete lists of physicochemical properties for DNA, RNA and protein sequences used in the stand-alone program are provided in **Table S4-S12**.

User-defined Physicochemical Properties

In the user-defined physicochemical index files, each index should be represented in three lines. The first line must start with asymbol (">") in the first column. The words right after the ">" symbol in the single initial line are optional and only used for the purpose of sequence identification and description of the index. The second line lists the names of the sequence compositions (i.e. amino acids, nucleotides, dinucleotides, or trinucleotides, etc), which should be sorted in the alphabet order, such as 'A' 'C' ... 'AA' 'AC'. All the elements in this line should be separated by TAB. The corresponding values of these sequence compositions are listed in the third line, which are separated by TAB.

For example, if you defined a physicochemical property "user_property", the user-defined physicochemical index file should be written as follows:

```
> user_property
A C ...    AA AC ...
0.21 0.12 ...    0.37 0.15 ...
```

After saving this file as "user_defined.txt" and specifying it using the command "-e user_defined.txt", or just "E user_defined.txt", the properties defined by user will be used in calculations.

Motif selection

There are two choices for input motif file by using command "-motif_database", one is "**ELM**" and another is "**Mega**", which means motif file comes from "ELM database" (13) and "MegaMotifBase" (14). For example, you can use the command "-motif_database ELM –motif your_motif_file.txt" to input motif information and calculate corresponding feature.

Output

The output file formats support three choices that are suitable for downstream computational analyses, such as machine learning. The first and the default choice is the tab format. In this format, all data is separated by TABs. The second one is the LIBSVM's sparse data format. For this format, each line contains an instance and is ended by a '\n' character, like <label> <index1>:<value1> <index2>:<value2> .... The <label> is a category label of the sequence. The pair <index>:<value> gives a feature (attribute) value: <index> is an integer starting from 1 and <value> is a real number. The third and fourth output format are the csv format and tsv format, which are similar to the tab format.

3.3 Scripts for results analysis

3.3.1 Method for results analysis

To obtain better feature vectors for constructing predictor, conducting feature analysis on feature generated by feature engineering might be an advisable choice. Here **BioSeq-BLM** provides a systematic result analysis framework including

normalization, **clustering**, **feature selection** and **dimension reduction**. Detailed methods and descriptions are listed in **Table S1**.

3.3.2 Script

"**FeatureAnalysis.py**" is an executive Python scripts used for feature analysis.

Input

The input file for "FeatureAnalysis.py" should be a feature vector file and its format belong to one of output format of feature extraction (tab, svm, csv or tsv).

Output

The output files also include the output results corresponding to the selected feature analysis method. For example, if you select a method in clustering, the clustering visualization results and clustering textfile will be generated.

3.4 Scripts for machine learning algorithms

Here we choose some machine learning algorithm commonly used like "**Support Vector Machine**" and deep learning model primarily used in the field of natural language processing like "**Transformer**" to construct predictor.

3.4.1 Support Vector Machine

Support Vector Machine (SVM) is a supervised learning algorithm that conducts data analysis for classification and regression (15,16). Here, the scikit-learn (17) package was used as the implementation of SVM algorithm with radial basis function as the kernel.

3.4.2 Random Forest

Random Forest (RF) is an ensemble learning method for classification, regression and some other tasks. In **BioSeq-BLM**, the RF algorithm in scikit-learn (17), a widely used machine learning Python package, was used as the implementation of RF algorithm.

3.4.3 Conditional Random Field

In order to capture the global information of residues for a long sequence, a sequence labelling algorithm Conditional Random Field (CRF) was provided for residue-level analysis. Compared with the traditional classification classifiers, such as SVM and RF, CRF is a sequence labelling algorithm that models the biological sequences in a global fashion and considering the dependency information of all the residues along the sequences (18).

3.4.4 Convolution Neural Network

In natural language processing, due to its high degree of parallelization, convolutional neural network (CNN) (19) is most commonly applied to the text classification problems. They are known as shift invariant or space invariant artificial neural networks based on their shared-weights architecture and translation invariance characteristics, which is capable of capturing a localized feature.

3.4.5 Long Short-Term Memory

Long short-term memory (LSTM) (20) is an artificial recurrent neural network (RNN) architecture. A common LSTM unit is composed of an input gate, an output gate and a forget gate, which makes it suitable for capturing long-term dependence feature than other convolutional neural networks.

3.4.6 Gated Recurrent Units

Gated recurrent units (GRU) (21) are a gating mechanism in recurrent neural networks (RNN). Different from the LSTM, there are only update gate and reset gate in GRU unit, whose advantages are reducing parameters and solving the problem of gradient disappearance in back propagation.

3.4.7 Transformer

Like recurrent neural networks (RNNs), Transformer (22) is designed to handle sequential data, especially for the natural language tasks, such as translation and text summarization. Based on self-attention mechanism and encoder-decoder architecture, the transformer models the association between any two units in the sequence and achieves the state-of-the-art performance in many NLP tasks. Transformers have become the primary choice for tackling many NLP problems, replacing most of recurrent neural network models, such as the long short-term memory (LSTM).

3.4.8 Weighted Transformer

Weighted Transformer, a Transformer with modified attention layers, replaces the multi-head attention by multiple self-attention branches learning to combine during the training process. Experimental verification indicates the weighted Transformer not only outperforms the baseline network, but also converges faster (23).

3.4.9 Reformer

Similar with Weighted Transformer, Reformer is an attention-based model improving Transformer. In the Reformer, the dot-product attention and the reversible residual layers are used to replace the locality-sensitive hashing attention and the standard residual layer, respectively. Reformer outperforms Transformer models. Reformer is much more memory-efficient and much faster on long sequence (24).

The parameters of above machine learning algorithms can be automatically optimized according to specific performance measures, such as accuracy (Acc), Balanced accuracy (BAcc), Matthew's correlation coefficient (MCC), area under ROC curve (AUC) or F1-score.

Detailed method, description and task application are listed in **Table S2**. In addition, to solve imbalanced dataset, we provide multiple sampling techniques. Details are listed in **Table S3**.

3.4.10 Scripts

"**MachineLearningRes.py**"and "**MachineLearningSeq.py**" are two executive Python scripts used for training predictors and evaluating their performance based on the input benchmark datasets. There are three main processes, including parameter selection, model training and cross validation, while the parameter selection process is mainly for "**Support Vector Machine**" and "**Random Forest**". In the parameter selection process, the parameters of machine learning algorithm are optimized on the validation sets. In this process, the multiprocessing technique is employed to significantly reduce the computational cost. Finally, in the cross-validation process, the performance of the constructed predictors is evaluated by k-fold cross-validation, jackknife or independent dataset test which can be selected by users.

Input

The input files of "**MachineLearningRes.py**" are two files of feature vectors in the format mention above generated by the feature extraction or feature analysis for "**Support Vector Machine**", "**Random Forest**" and "**Conditional Random Field**" or a feature vector file for deep learning model like "**Transformer**".

The input files of "**MachineLearningSeq.py**" are at least two files of feature vectors in the format mentioned above generated by the feature extraction or feature analysis. For binary classification problem, there should be two input files, storing the positive samples and the negative samples, respectively. For multiclass classification, at least three files are needed.

Output

The output files include trained model, ROC curve, PR curve and values of evaluation metrics which reprensent the performance of precictor.

3.5 Command

3.5.1 FeatureExtractionRes.py

Command line arguments for "FeatureExtractionRes.py":

Required	Description
-category {DNA,RNA,Protein}	The category of input sequences.
-method	Please select feature extraction method for residue level analysis.
-seq_file	The input sequence file in FASTA format.
-label_file	The corresponding label file.
Optional	**Description**

-h, --help	Show this help message and exit.
-trans {0,1}	Select whether use sliding window technique to transform sequence-labelling question to classification question.
-window WINDOW	The window size when construct sliding window technique for allocating every label a short sequence.
-fragment {0,1}	Please choose whether use the fragment method, 1 is yes while 0 is no.
-cpu CPU	The maximum number of CPU cores used for multiprocessing in generating frequency profile.
-pp_file PP_FILE	The physicochemical properties file user input. If input nothing, the default physicochemical properties is: DNA dinucleotide: Rise, Roll, Shift, Slide, Tilt, Twist. DNA trinucleotide: Dnase I, Bendability (DNAse). RNA: Rise, Roll, Shift, Slide, Tilt, Twist. Protein: Hydrophobicity, Hydrophilicity, Mass.
-fixed_len FIXED_LEN	The length of sequence will be fixed via cutting or padding. If you don't set value for 'fixed_len', it will be the maximum length of all input sequences.
-format {tab,svm,csv,tsv}	The output format (default=csv). tab – Simple format, delimited by TAB. svm -- The libSVM training data format. csv, tsv -- The format that can be loaded into a spreadsheet program.
-bp {0,1}	Select use batch mode or not, the parameter will change the directory for generating file based on the method you choose.

3.5.2 FeatureExtractionSeq.py

Command line arguments for “FeatureExtractionSeq.py”:

Required	Description
-category {DNA,RNA,Protein}	The category of input sequences.
-mode{OHE,BOW,TF-IDF, TR,WE,TM,SR,AF}	The feature extraction mode for input sequence which analogies with NLP, for

	example: bag of words (BOW).
-seq_file[SEQ_FILE [SEQ_FILE ...]]	The input file in FASTA format.
-label [LABEL [LABEL ...]]	The corresponding label of input sequence files.

Optional	Description
-h, --help	Show this help message and exit.
-words {Kmer,RevKmer,Mismatch, Subsequence,Top-NGram, DR,DT}	If you select mode in ['BOW', 'TF-IDF', 'TR', 'WE', 'TM'], you should select word for corresponding mode, for example Mismatch. Pay attention to that different category has different words, please reference to manual.
-method METHOD	If you select mode in ['OHE', 'WE', 'TM', 'SR', 'AF'], you should select method for corresponding mode, for example, select 'LDA' for 'TM' mode, select 'word2vec' for 'WE' mode and so on. For different category, the methods belong to 'OHE' and 'SR' mode is different, please reference to manual.
-auto_opt {0,1,2}	Choose whether automatically traverse the argument list. 2 is automatically traversing the argument list set ahead, 1 is automatically traversing the argument list in a smaller range, while 0 is not (default=0).
-cpu CPU	The maximum number of CPU cores used for multiprocessing in generating frequency profile and the number of CPU cores used for multiprocessing during parameter selection process (default=1).
-pp_file PP_FILE	The physicochemical properties file user input. If input nothing, the default physicochemical properties is: DNA dinucleotide: Rise, Roll, Shift, Slide, Tilt, Twist. DNA trinucleotide: Dnase I, Bendability (DNAse). RNA: Rise, Roll, Shift, Slide, Tilt, Twist. Protein: Hydrophobicity, Hydrophilicity, Mass.

-word_size[WORD_SIZE [WORD_SIZE ...]]	The word size of sequences for specific words (the range of word_size is between 1 and 6).
-mis_num[MIS_NUM [MIS_NUM ...]]	For Mismatch words. The max value inexact matching, mis_num should smaller than word_size (the range of mis_num is between 1 and 6).
-delta [DELTA [DELTA ...]]	For Subsequence words. The value of penalized factor (the range of delta is between 0 and 1).
-top_n [TOP_N [TOP_N ...]]	The maximum distance between structure statuses (the range of delta is between 1 and 4). It works with Top-n-gram words.
-max_dis[MAX_DIS [MAX_DIS ...]]	The max distance value for DR words and DT words (default is from 1 to 4).
-alpha ALPHA	Damping parameter for PageRank used in 'TR' mode, default=0.85.
-win_size WIN_SIZE	The maximum distance between the current and predicted word within a sentence for 'word2vec' in 'WE' mode, etc.
-vec_dim VEC_DIM	The output dimension of feature vectors for 'Glove' model and dimensionality of a word vectors for 'word2vec' and 'fastText' method.
-sg SG	Training algorithm for 'word2vec' and 'fastText' method. 1 for skip-gram, otherwise CBOW.
-in_tm{BOW,TF-IDF,TextRank}	While topic model implement subject extraction from a text, the text need to be preprocessed by one of mode in choices.
-com_prop COM_PROP	If choose topic model mode, please set component proportion for output feature vectors.
-oli {0,1}	Choose one kind of Oligonucleotide (default=0): 0 represents dinucleotid; 1 represents trinucleotide. For MAC, GAC, NMBAC methods of 'SR' mode.
-lag [LAG [LAG ...]]	The value of lag (default=1). For DACC, TACC, ACC, ACC-PSSM, AC-PSSM or CC-PSSM methods and so on.
-lamada[LAMADA [LAMADA ...]]	The value of lamada (default=1). For MAC, PDT, PDT-Profile, GAC or NMBAC methods

	and so on
-w [W [W ...]]	The value of weight (default=0.1). For ZCPseKNC method.
-k [K [K ...]]	The value of Kmer, it works only with ZCPseKNC method.
-n [N [N ...]]	The maximum distance between structure statuses (default=1). It works with PDT-Profile method.
-ui_file UI_FILE	The user-defined physicochemical property file.
-all_index	Choose all physicochemical indices.
-no_all_index	Do not choose all physicochemical indices, default.
-in_af	Choose the input for 'AF' mode from 'OHE' mode.
-lr LR	The value of learning rate, it works only with 'AF' mode.
-epochs EPOCHS	The epoch number of train process for 'AF' mode.
-batch_size BATCH_SIZE	The size of mini-batch, it works only with 'AF' mode.
-dropout DROPOUT	The value of dropout prob, it works only with 'AF' mode.
-fea_dim FEA_DIM	The output dimension of feature vectors, it works only with 'AF' mode.
-hidden_dim HIDDEN_DIM	The size of the intermediate (a.k.a., feed forward) layer, it works only with 'AF' mode.
-n_layer N_LAYER	The number of units for LSTM and GRU, it works only with 'AF' mode.
-motif_database {ELM,Mega}	The database where input motif file comes from.
-motif_file MOTIF_FILE	The short linear motifs from ELM database or structural motifs from the MegaMotifBase.
-score {ED,MD,CD,HD,JSC,CS, PCC,KLD,none}	Choose whether calculate semantic similarity score and what method for calculation.
-cv {5,10,j}	The cross validation mode. 5 or 10: 5-fold or 10-fold cross validation, j: (character 'j')

	jackknife cross validation.
-fixed_len FIXED_LEN	The length of sequence will be fixed via cutting orpadding. If you don't set value for 'fixed_len', it will be the maximum length of all input sequences.
-format {tab,svm,csv,tsv}	The output format (default=csv). tab – Simple format, delimited by TAB. svm -- The libSVM training data format. csv, tsv -- The format that can be loaded into a spreadsheet program.
-bp {0,1}	Select use batch mode or not, the parameter will change the directory for generating file based on the method you choose.

3.5.3 FeatureAnalysis.py

Command line arguments for "FeatureAnalysis.py":

Required	Description
-vec_file[VEC_FILE [VEC_FILE ...]]	The input feature vector files.
-label [LABEL [LABEL ...]]	The corresponding label of input vector files is required.

Optional	Description
-h, --help	Show this help message and exit.
-sn {min-max-scale,standard-scale, L1-normalize,L2-normalize,none}	Choose method of standardization or normalization for feature vectors.
-cl {AP,DBSCAN,GMM, AGNES,Kmeans,none}	Choose method for clustering.
-cm {feature,sample}	The mode for clustering.
-nc NC	The number of clusters.
-fs{chi2,F-value,MIC,RFE,Tree,none}	Select feature select method.
-nf NF	The number of features after feature selection.
-dr{PCA, KernelPCA,TSVD,none}	Choose method for dimension reduction.

-np NP	The dimension of main component after dimension reduction.
-rdb {no,fs,dr}	Reduce dimension by: 'no'---none; 'fs'---apply feature selection to parameter selection procedure; 'dr'--- apply dimension reduction to parameter selection procedure.
-format {tab,svm,csv,tsv}	The output format (default=csv). tab – Simple format, delimited by TAB. svm -- The libSVM training data format. csv, tsv -- The format that can be loaded into a spreadsheet program.

3.5.4 MachineLearningRes.py

Command line arguments for "MachineLearningRes.py":

Required	Description
-ml {SVM,RF,CRF,CNN,LSTM, GRU,Transformer,Weighted-Transformer,Reformer}	The machine-learning algorithm for constructing predictor, for example: Support Vector Machine (SVM).
-vec_file[VEC_FILE [VEC_FILE ...]]	The input feature vector file(s). If dichotomy, inputting positive sample before negative sample is required.
-label_file	The corresponding label file is required.
Optional	**Description**
-h, --help	Show this help message and exit.
-cpu CPU	The number of CPU cores used for multiprocessing during parameter selection process (default=1).
-grid [{0,1} [{0,1} ...]]	Grid=0 for rough grid search, grid=1 for meticulous grid search.
-cost [COST [COST ...]]	Regularization parameter of 'SVM'.
-gamma[GAMMA [GAMMA ...]]	Kernel coefficient for 'rbf' of 'SVM'.
-tree [TREE [TREE ...]]	The number of trees in the forest for 'RF'.

-lr LR	The value of learning rate for deep learning.
-epochs EPOCHS	The epoch number for training deep learning model.
-batch_size BATCH_SIZE	The size of mini-batch for deep learning.
-dropout DROPOUT	The value of dropout prob for deep learning.
-hidden_dim HIDDEN_DIM	The size of the intermediate (a.k.a., feed forward) layer.
-n_layer N_LAYER	The number of units for 'LSTM' and 'GRU'.
-out_channels OUT_CHANNELS	The number of output channels for 'CNN'
-kernel_size KERNEL_SIZE	The size of stride for 'CNN'.
-d_model D_MODEL	The dimension of multi-head attention layer for Transformer or Weighted-Transformer.
-d_ff D_FF	The dimension of fully connected layer of Transformer or Weighted-Transformer.
-heads HEADS	The number of heads for Transformer or Weighted-Transformer.
-metric {Acc,MCC,AUC,BAcc,F1}	The metric for parameter selection
-cv {5,10,j}	The cross validation mode. 5 or 10: 5-fold or 10-fold cross validation, j: (character 'j') jackknife cross validation.
-sp {none,over,under,combine}	Select technique for oversampling.
-ind_vec_file [IND_VEC_FILE [IND_VEC_FILE ...]]	The feature vector files of independent test dataset.
-ind_label_file IND_LABEL_FILE	The corresponding label file of independent test dataset.
-format {tab,svm,csv,tsv}	The input format (default=csv). tab – Simple format, delimited by TAB. svm -- The libSVM training data format. csv, tsv -- The format that can be loaded into a spreadsheet program.

3.5.5 MachineLearningSeq.py

Command line arguments for "MachineLearningSeq.py":

Required	Description
-ml {SVM,RF,CNN,LSTM, GRU,Transformer,Weighted-Transformer,Reformer}	The machine-learning algorithm for constructing predictor, for example: Support Vector Machine (SVM).
-vec_file[VEC_FILE [VEC_FILE ...]]	The input feature vector files.
-label [LABEL [LABEL ...]]	The corresponding label of input vector files is required.
Optional	**Description**
-h, --help	Show this help message and exit.
-cpu CPU	The number of CPU cores used for multiprocessing during parameter selection process (default=1).
-grid [{0,1} [{0,1} ...]]	Grid=0 for rough grid search, grid=1 for meticulous grid search.
-cost [COST [COST ...]]	Regularization parameter of 'SVM'.
-gamma[GAMMA [GAMMA ...]]	Kernel coefficient for 'rbf' of 'SVM'.
-tree [TREE [TREE ...]]	The number of trees in the forest for 'RF'.
-lr LR	The value of learning rate for deep learning.
-epochs EPOCHS	The epoch number for train deep model.
-batch_size BATCH_SIZE	The size of mini-batch for deep learning.
-dropout DROPOUT	The value of dropout prob for deep learning.
-hidden_dim HIDDEN_DIM	The size of the intermediate (a.k.a., feed forward) layer.
-n_layer N_LAYER	The number of units for 'LSTM' and 'GRU'.
-out_channels OUT_CHANNELS	The number of output channels for 'CNN'

-kernel_size KERNEL_SIZE	The size of stride for 'CNN'.
-d_model D_MODEL	The dimension of multi-head attention layer for Transformer or Weighted-Transformer.
-d_ff D_FF	The dimension of fully connected layer of Transformer or Weighted-Transformer.
-heads HEADS	The number of heads for Transformer or Weighted-Transformer.
-metric {Acc,MCC,AUC,BAcc,F1}	The metric for parameter selection
-cv {5,10,j}	The cross validation mode. 5 or 10: 5-fold or 10-fold cross validation, j: (character 'j') jackknife cross validation.
-sp {none,over,under,combine}	Select technique for oversampling.
-ind_vec_file [IND_VEC_FILE [IND_VEC_FILE ...]]	The feature vector files of independent test dataset.
-format {tab,svm,csv,tsv}	The input format (default=csv). tab – Simple format, delimited by TAB. svm -- The libSVM training data format. csv, tsv -- The format that can be loaded into a spreadsheet program.

3.5.6 BioSeq-BLM _Res.py

Command line arguments for “BioSeq-BLM_Res.py”:

Required	Description
-category {DNA,RNA,Protein}	The category of input sequences.
-method	Please select feature extraction method for residue level analysis.
-ml {SVM,RF,CRF,CNN,LSTM, GRU,Transformer,Weighted-Transformer,Reformer}	The machine-learning algorithm for constructing predictor, for example: Support Vector Machine (SVM).
-seq_file	The input file in FASTA format.
-label_file	The corresponding label file.

Optional	Description

-h, --help	Show this help message and exit.
-trans {0,1}	Select whether use sliding window technique to transform sequence-labelling question to classification question.
-window WINDOW	The window size when construct sliding window technique for allocating every label a short sequence.
-fragment {0,1}	Please choose whether use the fragment method, 1 is yes while 0 is no.
-cpu CPU	The maximum number of CPU cores used for multiprocessing in generating frequency profile or The number of CPU cores used for multiprocessing during parameter selection process (default=1).
-pp_file PP_FILE	The physicochemical properties file user input. If input nothing, the default physicochemical properties is: DNA dinucleotide: Rise, Roll, Shift, Slide, Tilt, Twist. DNA trinucleotide: Dnase I, Bendability (DNAse). RNA: Rise, Roll, Shift, Slide, Tilt, Twist. Protein: Hydrophobicity, Hydrophilicity, Mass.
-sn {min-max-scale,standard-scale, L1-normalize,L2-normalize,none}	Choose method of standardization or normalization for feature vectors.
-cl {AP,DBSCAN,GMM, AGNES,Kmeans,none}	Choose method for clustering.
-cm {feature,sample}	The mode for clustering.
-nc NC	The number of clusters.
-fs{chi2,F-value,MIC,RFE,Tree,none}	Select feature select method.
-nf NF	The number of features after feature selection.
-dr{PCA, KernelPCA,TSVD,none}	Choose method for dimension reduction.
-np NP	The dimension of main component after dimension reduction.
-rdb {no,fs,dr}	Reduce dimension by: 'no'---none; 'fs'---apply feature selection to parameter

	selection procedure; 'dr'--- apply dimension reduction to parameter selection procedure.
-grid [{0,1} [{0,1} ...]]	Grid=0 for rough grid search, grid=1 for meticulous grid search.
-cost [COST [COST ...]]	Regularization parameter of 'SVM'.
-gamma[GAMMA [GAMMA ...]]	Kernel coefficient for 'rbf' of 'SVM'.
-tree [TREE [TREE ...]]	The number of trees in the forest for 'RF'.
-lr LR	The value of learning rate for deep learning.
-epochs EPOCHS	The epoch number for train deep model.
-batch_size BATCH_SIZE	The size of mini-batch for deep learning.
-dropout DROPOUT	The value of dropout prob for deep learning.
-hidden_dim HIDDEN_DIM	The size of the intermediate (a.k.a., feed forward) layer.
-n_layer N_LAYER	The number of units for 'LSTM' and 'GRU'.
-out_channels OUT_CHANNELS	The number of output channels for 'CNN'
-kernel_size KERNEL_SIZE	The size of stride for 'CNN'.
-d_model D_MODEL	The dimension of multi-head attention layer for Transformer or Weighted-Transformer.
-d_ff D_FF	The dimension of fully connected layer of Transformer or Weighted-Transformer.
-heads HEADS	The number of heads for Transformer or Weighted-Transformer.
-metric {Acc,MCC,AUC,BAcc,F1}	The metric for parameter selection
-cv {5,10,j}	The cross validation mode. 5 or 10: 5-fold or 10-fold cross validation, j: (character 'j') jackknife cross validation.

-sp {none,over,under,combine}	Select technique for oversampling.
-ind_seq_file IND_SEQ_FILE	The independent test dataset in FASTA format.
-ind_label_file IND_LABEL_FILE	The corresponding label file of independent test dataset.
-fixed_len FIXED_LEN	The length of sequence will be fixed via cutting orpadding. If you don't set value for 'fixed_len', it will be the maximum length of all input sequences.
-format {tab,svm,csv,tsv}	The output format (default=csv). tab – Simple format, delimited by TAB. svm -- The libSVM training data format. csv, tsv -- The format that can be loaded into a spreadsheet program.
-bp {0,1}	Select use batch mode or not, the parameter will change the directory for generating file based on the method you choose.

3.5.7 BioSeq-BLM_Seq.py

Command line arguments for "BioSeq-BLM_Seq.py":

Required	Description
-category {DNA,RNA,Protein}	The category of input sequences.
-mode{OHE,BOW,TF-IDF,TR,WE, TM,SR,AF}	The feature extraction mode for input sequence which analogies with NLP, for example: bag of words (BOW).
-ml {SVM,RF,CNN,LSTM, GRU,Transformer,Weighted-Transformer,Reformer}	The machine-learning algorithm for constructing predictor, for example: Support Vector Machine (SVM).
-seq_file[SEQ_FILE [SEQ_FILE ...]]	The input file in FASTA format.
-label [LABEL [LABEL ...]]	The corresponding label of input sequence files.
Optional	**Description**
-h, --help	Show this help message and exit.

-score {ED,MD,CD,HD,JSC,CS, PCC,KLD,none}	Choose whether calculate semantic similarity score and what method for calculation.
-words {Kmer,RevKmer,Mismatch, Subsequence,Top-NGram, DR,DT}	If you select mode in ['BOW', 'TF-IDF', 'TR', 'WE', 'TM'], you should select word for corresponding mode, for example Mismatch. Pay attention to that different category has different words, please reference to manual.
-method METHOD	If you select mode in ['OHE', 'WE', 'TM', 'SR', 'AF'], you should select method for corresponding mode, for example, select 'LDA' for 'TM' mode, select 'word2vec' for 'WE' mode and so on. For different category, the methods belong to 'OHE' and 'SR' mode is different, please reference to manual.
-auto_opt {0,1,2}	Choose whether automatically traverse the argument list. 2 is automatically traversing the argument list set ahead, 1 is automatically traversing the argument list in a smaller range, while 0 is not (default=0).
-cpu CPU	The maximum number of CPU cores used for multiprocessing in generating frequency profile and the number of CPU cores used for multiprocessing during parameter selection process (default=1).
-pp_file PP_FILE	The physicochemical properties file user input. If input nothing, the default physicochemical properties is: DNA dinucleotide: Rise, Roll, Shift, Slide, Tilt, Twist. DNA trinucleotide: Dnase I, Bendability (DNAse). RNA: Rise, Roll, Shift, Slide, Tilt, Twist. Protein: Hydrophobicity, Hydrophilicity, Mass.
-word_size[WORD_SIZE [WORD_SIZE ...]]	The word size of sequences for specific words (the range of word_size is between 1 and 6).
-mis_num[MIS_NUM [MIS_NUM ...]]	For Mismatch words. The max value inexact matching, mis_num should smaller than word_size (the range of mis_num is

	between 1 and 6).
-delta [DELTA [DELTA ...]]	For Subsequence words. The value of penalized factor (the range of delta is between 0 and 1).
-top_n [TOP_N [TOP_N ...]]	The maximum distance between structure statuses (the range of delta is between 1 and 4). It works with Top-n-gram words.
-max_dis[MAX_DIS [MAX_DIS ...]]	The max distance value for DR words and DT words (default is from 1 to 4).
-alpha ALPHA	Damping parameter for PageRank used in 'TR' mode, default=0.85.
-win_size WIN_SIZE	The maximum distance between the current and predicted word within a sentence for 'word2vec' in 'WE' mode, etc.
-vec_dim VEC_DIM	The output dimension of feature vectors for 'Glove' model and dimensionality of a word vectors for 'word2vec' and 'fastText' method.
-sg SG	Training algorithm for 'word2vec' and 'fastText' method. 1 for skip-gram, otherwise CBOW.
-in_tm{BOW,TF-IDF,TextRank}	While topic model implement subject extraction from a text, the text need to be preprocessed by one of mode in choices.
-com_prop COM_PROP	If choose topic model mode, please set component proportion for output feature vectors.
-oli {0,1}	Choose one kind of Oligonucleotide (default=0): 0 represents dinucleotid; 1 represents trinucleotide. For MAC, GAC, NMBAC methods of 'SR' mode.
-lag [LAG [LAG ...]]	The value of lag (default=1). For DACC, TACC, ACC, ACC-PSSM, AC-PSSM or CC-PSSM methods and so on.
-lamada[LAMADA [LAMADA ...]]	The value of lamada (default=1). For MAC, PDT, PDT-Profile, GAC or NMBAC methods and so on
-w [W [W ...]]	The value of weight (default=0.1). For ZCPseKNC method.

-k [K [K ...]]	The value of Kmer, it works only with ZCPseKNC method.
-n [N [N ...]]	The maximum distance between structure statuses (default=1). It works with PDT-Profile method.
-ui_file UI_FILE	The user-defined physicochemical property file.
-all_index	Choose all physicochemical indices.
-no_all_index	Do not choose all physicochemical indices, default.
-in_af	Choose the input for 'AF' mode from 'OHE' mode.
-fea_dim FEA_DIM	The output dimension of feature vectors, it works only with 'AF' mode.
-motif_database {ELM,Mega}	The database where input motif file comes from.
-motif_file MOTIF_FILE	The short linear motifs from ELM database or structural motifs from the MegaMotifBase.
-sn {min-max-scale,standard-scale, L1-normalize,L2-normalize,none}	Choose method of standardization or normalization for feature vectors.
-cl {AP,DBSCAN,GMM, AGNES,Kmeans,none}	Choose method for clustering.
-cm {feature,sample}	The mode for clustering.
-nc NC	The number of clusters.
-fs{chi2,F-value,MIC,RFE,Tree,none}	Select feature select method.
-nf NF	The number of features after feature selection.
-dr{PCA, KernelPCA,TSVD,none}	Choose method for dimension reduction.
-np NP	The dimension of main component after dimension reduction.
-rdb {no,fs,dr}	Reduce dimension by: 'no'---none; 'fs'---apply feature selection to parameter selection procedure; 'dr'--- apply dimension reduction to parameter selection

	procedure.
-grid [{0,1} [{0,1} ...]]	Grid=0 for rough grid search, grid=1 for meticulous grid search.
-cost [COST [COST ...]]	Regularization parameter of 'SVM'.
-gamma[GAMMA [GAMMA ...]]	Kernel coefficient for 'rbf' of 'SVM'.
-tree [TREE [TREE ...]]	The number of trees in the forest for 'RF'.
-lr LR	The value of learning rate for 'AF' mode and deep learning.
-epochs EPOCHS	The epoch number for train deep model.
-batch_size BATCH_SIZE	The size of mini-batch for 'AF' mode and deep learning.
-dropout DROPOUT	The value of dropout prob for 'AF' mode and deep learning.
-hidden_dim HIDDEN_DIM	The size of the intermediate (a.k.a., feed forward) layer.
-n_layer N_LAYER	The number of units for 'LSTM' and 'GRU'.
-out_channels OUT_CHANNELS	The number of output channels for 'CNN'
-kernel_size KERNEL_SIZE	The size of stride for 'CNN'.
-d_model D_MODEL	The dimension of multi-head attention layer for Transformer or Weighted-Transformer.
-d_ff D_FF	The dimension of fully connected layer of Transformer or Weighted-Transformer.
-heads HEADS	The number of heads for Transformer or Weighted-Transformer.
-metric {Acc,MCC,AUC,BAcc,F1}	The metric for parameter selection
-cv {5,10,j}	The cross validation mode. 5 or 10: 5-fold or 10-fold cross validation, j: (character 'j') jackknife cross validation.
-sp {none,over,under,combine}	Select technique for oversampling.

-ind_seq_file [IND_SEQ_FILE [IND_SEQ_FILE ...]]	The independent test dataset in FASTA format.
-fixed_len FIXED_LEN	The length of sequence will be fixed via cutting orpadding. If you don't set value for 'fixed_len', it will be the maximum length of all input sequences.
-format {tab,svm,csv,tsv}	The output format (default=csv). tab – Simple format, delimited by TAB. svm -- The libSVM training data format. csv, tsv -- The format that can be loaded into a spreadsheet program.
-bp {0,1}c	Select use batch mode or not, the parameter will change the directory for generating file based on the method you choose.

4 Tables

Table S1. Feature analysis for Biological sequences.

Algorithm	Method	Description
Standardization or Normalization	min-max-scale	Normalization by scikit-learn (4) 'MinMaxScaler'
	standard-scale	Standardization by scikit-learn (4) 'StandardScaler'
	L1-normalize	Normalization based on L1 regularization (25)
	L2-normalize	Normalization based on L2 regularization (26)
Clustering	AP	Clustering based on Affinity Propagation algorithm (27)
	DBSCAN	Clustering based on Density-Based Spatial Clustering of Applications with Noise algorithm (28)
	GMM	Clustering based on Gaussian Mixture Model (29)
	AGNES	Clustering based on agglomerative nesting algorithm (30)
	K-means	Clustering based on K-means algorithm (31)

Feature selection	chi2	Univariate feature selection based on Chi-square test (32,33)
	F-value	Univariate feature selection based on F-test (joint hypotheses test) (32,33)
	MIC	Univariate feature selection with mutual information (32,33)
	RFE	Select feature based on Recursive Feature Elimination (34)
	Tree	Tree-based feature selection (35)
Dimension reduction	PCA	Reduce dimension based on principal component analysis (36),
	KernelPCA	Reduce dimension based on principal component analysis with 'rbf' kernel (37)
	TSVD	Reduce dimension based on truncated singular value decomposition (38)

Table S2. Machine learning algorithm for constructing predictor.

Category	Method	Description	Analysis Level*
classification algorithm	SVM	Support Vector Machine (39)	S, R
	RF	Random Forest (40)	
sequence labelling algorithm	CRF	Conditional Random Field (41)	R
Deep learning algorithm	CNN	Convolutional Neural Networks (42)18	S, R
	LSTM	Long Short-Term Memory (20)	
	GRU	Gate Recurrent Unit (21)	
	Transformer	Network completely based on self-attention (22)	
	Weighted Transformer	Weighted Transformer network (23)	
	Reformer	Efficient Transformer (24)	

***** S for sequence level, R for residue level.

Table S3. Sampling technique for constructing predictor.

Method	Description
over	Over-sampling based on Synthetic Minority Oversampling Technique (SMOTE) (43)
under	Under-sampling based on Tomek links method (44)
combine	Combine over-sampling and under-sampling by 'SMOTETomek' in sklearn package (45)

Table S4. The names of the 148 physicochemical indices for dinucleotides.

Base stacking	Protein induced deformability	B-DNA twist
Propeller twist	Duplex stability:(freeenergy)	Duplex tability(disruptenergy)
Protein DNA twist	Stabilising energy of Z-DNA	Aida_BA_transition
Breslauer_dS	Electron_interaction	Hartman_trans_free_energy
Lisser_BZ_transition	Polar_interaction	SantaLucia_dG
Sarai_flexibility	Stability	Stacking_energy
Sugimoto_dS	Watson-Crick_interaction	Twist
Shift	Slide	Rise
Twist stiffness	Tilt stiffness	Shift_rise
Twist_shift	Enthalpy1	Twist_twist
Shift2	Tilt3	Tilt1
Slide (DNA-protein complex)1	Tilt_shift	Twist_tilt
Roll_rise	Stacking energy	Stacking energy1
Propeller Twist	Roll11	Rise (DNA-protein complex)
Roll2	Roll3	Roll1
Slide_slide	Enthalpy	Shift_shift
Flexibility_slide	Minor Groove Distance	Rise (DNA-protein complex)1
Roll (DNA-protein complex)1	Entropy	Cytosine content

Major Groove Distance	Twist (DNA-protein complex)	Purine (AG) content
Tilt_slide	Major Groove Width	Major Groove Depth
Free energy6	Free energy7	Free energy4
Free energy3	Free energy1	Twist_roll
Flexibility_shift	Shift (DNA-protein complex)1	Thymine content
Tip	Keto (GT) content	Roll stiffness
Entropy1	Roll_slide	Slide (DNA-protein complex)
Twist2	Twist5	Twist4
Tilt (DNA-protein complex)1	Twist_slide	Minor Groove Depth
Persistance Length	Rise3	Shift stiffness
Slide3	Slide2	Slide1
Rise1	Rise stiffness	Mobility to bend towards minor groove
Dinucleotide GC Content	A-philicity	Wedge
DNA denaturation	Bending stiffness	Free energy5
Breslauer_dG	Breslauer_dH	Shift (DNA-protein complex)
Helix-Coil_transition	Ivanov_BA_transition	Slide_rise
SantaLucia_dH	SantaLucia_dS	Minor Groove Width
Sugimoto_dG	Sugimoto_dH	Twist1
Tilt	Roll	Twist7
Clash Strength	Roll_roll	Roll (DNA-protein complex)
Adenine content	Direction	Probability contacting nucleosome core
Roll_shift	Shift_slide	Shift1
Tilt4	Tilt2	Free energy8
Twist (DNA-protein complex)1	Tilt_rise	Free energy2
Stacking energy2	Stacking energy3	Rise_rise
Tilt_tilt	Roll4	Tilt_roll

Minor Groove Size	GC content	Inclination
Slide stiffness	Melting Temperature1	Twist3
Tilt (DNA-protein complex)	Guanine content	Twist6
Major Groove Size	Twist_rise	Rise2
Melting Temperature	Free energy	Mobility to bend towards major groove
Bend		

Table S5. The names of the 12 physicochemical indices for trinucleotides.

Bendability (DNAse)	Bendability (consensus)	Trinucleotide GC Content
Consensus_roll	Consensus-Rigid	Dnase I
MW-Daltons	MW-kg	Nucleosome
Nucleosome positioning	Dnase I-Rigid	Nucleosome-Rigid

Table S6. The names of the 90 physicochemical indices for dinucleotides.

Base stacking	Protein induced deformability	B-DNA twist
Dinucleotide GC Content	A-philicity	Propeller twist
Duplex stability-free energy	Duplex stability-disrupt energy	DNA denaturation
Bending stiffness	Protein DNA twist	Stabilising energy of Z-DNA
Aida_BA_transition	Breslauer_dG	Breslauer_dH
Breslauer_dS	Electron_interaction	Hartman_trans_free_energy
Helix-Coil_transition	Ivanov_BA_transition	Lisser_BZ_transition
Polar_interaction	SantaLucia_dG	SantaLucia_dH
SantaLucia_dS	Sarai_flexibility	Stability
Stacking_energy	Sugimoto_dG	Sugimoto_dH
Sugimoto_dS	Watson-Crick_interaction	Twist

Tilt	Roll	Shift
Slide	Rise	Stacking energy
Bend	Tip	Inclination
Major Groove Width	Major Groove Depth	Major Groove Size
Major Groove Distance	Minor Groove Width	Minor Groove Depth
Minor Groove Size	Minor Groove Distance	Persistance Length
Melting Temperature	Mobility to bend towards major groove	Mobility to bend towards minor groove
Propeller Twist	Clash Strength	Enthalpy
Free energy	Twist_twist	Tilt_tilt
Roll_roll	Twist_tilt	Twist_roll
Tilt_roll	Shift_shift	Slide_slide
Rise_rise	Shift_slide	Shift_rise
Slide_rise	Twist_shift	Twist_slide
Twist_rise	Tilt_shift	Tilt_slide
Tilt_rise	Roll_shift	Roll_slide
Roll_rise	Slide stiffness	Shift stiffness
Roll stiffness	Rise stiffness	Tilt stiffness
Twist stiffness	Wedge	Direction
Flexibility_slide	Flexibility_shift	Entropy

Table S7. The names of the six physicochemical indices for dinucleotides.

Twist	Tilt	Roll
Shift	Slide	Rise

Table S8. The names of the 22 physicochemical indices for dinucleotides.

Shift (RNA)	Hydrophilicity (RNA)
Hydrophilicity (RNA)	GC content
Purine (AG) content	Keto (GT) content

Adenine content	Guanine content
Cytosine content	Thymine content
Slide (RNA)	Rise (RNA)
Tilt (RNA)	Roll (RNA)
Twist (RNA)	Stacking energy (RNA)
Enthalpy (RNA)	Entropy (RNA)
Free energy (RNA)	Free energy (RNA)
Enthalpy (RNA)	Entropy (RNA)

Table S9. The names of the 11 physicochemical indices for dinucleotides.

Shift	Slide	Rise
Tilt	Roll	Twist
Stacking energy	Enthalpy	Entropy
Free energy	Hydrophilicity	

Table S10. The names of the 547 physicochemical indices for amino acids.

Hydrophobicity	Hydrophilicity	Mass
ARGP820102	ARGP820103	BEGF750101
BHAR880101	BIGC670101	BIOV880101
BROC820102	BULH740101	BULH740102
BUNA790103	BURA740101	BURA740102
CHAM820102	CHAM830101	CHAM830102
CHAM830105	CHAM830106	CHAM830107
CHOC760101	CHOC760102	CHOC760103
CHOP780201	CHOP780202	CHOP780203
CHOP780206	CHOP780207	CHOP780208
CHOP780211	CHOP780212	CHOP780213
CHOP780216	CIDH920101	CIDH920102
CIDH920105	COHE430101	CRAJ730101

DAWD720101	DAYM780101	DAYM780201
EISD840101	EISD860101	EISD860102
FASG760102	FASG760103	FASG760104
FAUJ880101	FAUJ880102	FAUJ880103
FAUJ880106	FAUJ880107	FAUJ880108
FAUJ880111	FAUJ880112	FAUJ880113
FINA910102	FINA910103	FINA910104
GEIM800102	GEIM800103	GEIM800104
GEIM800107	GEIM800108	GEIM800109
GOLD730101	GOLD730102	GRAR740101
GUYH850101	HOPA770101	HOPT810101
HUTJ700103	ISOY800101	ISOY800102
ISOY800105	ISOY800106	ISOY800107
JANJ780102	JANJ780103	JANJ790101
JOND750102	JOND920101	JOND920102
KANM800101	KANM800102	KANM800103
KARP850102	KARP850103	KHAG800101
KRIW790101	KRIW790102	KRIW790103
LEVM760101	LEVM760102	LEVM760103
LEVM760106	LEVM760107	LEVM780101
LEVM780104	LEVM780105	LEVM780106
LIFS790102	LIFS790103	MANP780101
MAXF760103	MAXF760104	MAXF760105
MEEJ800101	MEEJ800102	MEEJ810101
MEIH800102	MEIH800103	MIYS850101
NAGK730103	NAKH900101	NAKH900102
NAKH900105	NAKH900106	NAKH900107
NAKH900110	NAKH900111	NAKH900112

NAKH920102	NAKH920103	NAKH920104
NAKH920107	NAKH920108	NISK800101
OOBM770101	OOBM770102	OOBM770103
OOBM850101	OOBM850102	OOBM850103
PALJ810101	PALJ810102	PALJ810103
PALJ810106	PALJ810107	PALJ810108
PALJ810111	PALJ810112	PALJ810113
PALJ810116	PARJ860101	PLIV810101
PONP800103	PONP800104	PONP800105
PONP800108	PRAM820101	PRAM820102
PRAM900102	PRAM900103	PRAM900104
QIAN880101	QIAN880102	QIAN880103
QIAN880106	QIAN880107	QIAN880108
QIAN880111	QIAN880112	QIAN880113
QIAN880116	QIAN880117	QIAN880118
QIAN880121	QIAN880122	QIAN880123
QIAN880126	QIAN880127	QIAN880128
QIAN880131	QIAN880132	QIAN880133
QIAN880136	QIAN880137	QIAN880138
RACS770102	RACS770103	RACS820101
RACS820104	RACS820105	RACS820106
RACS820109	RACS820110	RACS820111
RACS820114	RADA880101	RADA880102
RADA880105	RADA880106	RADA880107
RICJ880102	RICJ880103	RICJ880104
RICJ880107	RICJ880108	RICJ880109
RICJ880112	RICJ880113	RICJ880114
RICJ880117	ROBB760101	ROBB760102

ROBB760105	ROBB760106	ROBB760107
ROBB760110	ROBB760111	ROBB760112
ROSG850101	ROSG850102	ROSM880101
SIMZ760101	SNEP660101	SNEP660102
SUEM840101	SUEM840102	SWER830101
TANS770103	TANS770104	TANS770105
TANS770108	TANS770109	TANS770110
VASM830103	VELV850101	VENT840101
WEBA780101	WERD780101	WERD780102
WOEC730101	WOLR810101	WOLS870101
YUTK870101	YUTK870102	YUTK870103
ZIMJ680101	ZIMJ680102	ZIMJ680103
AURR980101	AURR980102	AURR980103
AURR980106	AURR980107	AURR980108
AURR980111	AURR980112	AURR980113
AURR980116	AURR980117	AURR980118
ONEK900101	ONEK900102	VINM940101
VINM940104	MUNV940101	MUNV940102
MUNV940105	WIMW960101	KIMC930101
PARS000101	PARS000102	KUMS000101
KUMS000104	TAKK010101	FODM020101
NADH010103	NADH010104	NADH010105
MONM990201	KOEP990101	KOEP990102
CEDJ970103	CEDJ970104	CEDJ970105
FUKS010103	FUKS010104	FUKS010105
FUKS010108	FUKS010109	FUKS010110
AVBF000101	AVBF000102	AVBF000103
AVBF000106	AVBF000107	AVBF000108

MITS020101	TSAJ990101	TSAJ990102
WILM950101	WILM950102	WILM950103
GUOD860101	JURD980101	BASU050101
SUYM030101	PUNT030101	PUNT030102
GEOR030103	GEOR030104	GEOR030105
GEOR030108	GEOR030109	ZHOH040101
BAEK050101	HARY940101	PONJ960101
OLSK800101	KIDA850101	GUYH850102
GUYH850105	ROSM880104	ROSM880105
BLAS910101	CASG920101	CORJ870101
CORJ870104	CORJ870105	CORJ870106
MIYS990101	MIYS990102	MIYS990103
ENGD860101	FASG890101	TANS770101
ANDN920101	ARGP820101	TANS770106
BEGF750102	BEGF750103	VASM830101
BIOV880102	BROC820101	VHEG790101
BUNA790101	BUNA790102	WERD780103
CHAM810101	CHAM820101	WOLS870102
CHAM830103	CHAM830104	YUTK870104
CHAM830108	CHOC750101	ZIMJ680104
CHOC760104	CHOP780101	AURR980104
CHOP780204	CHOP780205	AURR980109
CHOP780209	CHOP780210	AURR980114
CHOP780214	CHOP780215	AURR980119
CIDH920103	CIDH920104	VINM940102
CRAJ730102	CRAJ730103	MUNV940103
DESM900101	DESM900102	MONM990101
EISD860103	FASG760101	KUMS000102

FASG760105	FAUJ830101	NADH010101
FAUJ880104	FAUJ880105	NADH010106
FAUJ880109	FAUJ880110	CEDJ970101
FINA770101	FINA910101	FUKS010101
GARJ730101	GEIM800101	FUKS010106
GEIM800105	GEIM800106	FUKS010111
GEIM800110	GEIM800111	AVBF000104
GRAR740102	GRAR740103	AVBF000109
HUTJ700101	HUTJ700102	COSI940101
ISOY800103	ISOY800104	WILM950104
ISOY800108	JANJ780101	BASU050102
JANJ790102	JOND750101	GEOR030101
JUKT750101	JUNJ780101	GEOR030106
KANM800104	KARP850101	ZHOH040102
KLEP840101	KRIW710101	DIGM050101
KYTJ820101	LAWE840101	GUYH850103
LEVM760104	LEVM760105	JACR890101
LEVM780102	LEVM780103	CORJ870102
LEWP710101	LIFS790101	CORJ870107
MAXF760101	MAXF760102	MIYS990104
MAXF760106	MCMT640101	TANS770102
MEEJ810102	MEIH800101	TANS770107
NAGK730101	NAGK730102	VASM830102
NAKH900103	NAKH900104	WARP780101
NAKH900108	NAKH900109	WERD780104
NAKH900113	NAKH920101	WOLS870103
NAKH920105	NAKH920106	ZASB820101
NISK860101	NOZY710101	ZIMJ680105

OOBM770104	OOBM770105	AURR980105
OOBM850104	OOBM850105	AURR980110
PALJ810104	PALJ810105	AURR980115
PALJ810109	PALJ810110	AURR980120
PALJ810114	PALJ810115	VINM940103
PONP800101	PONP800102	MUNV940104
PONP800106	PONP800107	BLAM930101
PRAM820103	PRAM900101	KUMS000103
PTIO830101	PTIO830102	NADH010102
QIAN880104	QIAN880105	NADH010107
QIAN880109	QIAN880110	CEDJ970102
QIAN880114	QIAN880115	FUKS010102
QIAN880119	QIAN880120	FUKS010107
QIAN880124	QIAN880125	FUKS010112
QIAN880129	QIAN880130	AVBF000105
QIAN880134	QIAN880135	YANJ020101
QIAN880139	RACS770101	PONP930101
RACS820102	RACS820103	KUHL950101
RACS820107	RACS820108	BASU050103
RACS820112	RACS820113	GEOR030102
RADA880103	RADA880104	GEOR030107
RADA880108	RICJ880101	ZHOH040103
RICJ880105	RICJ880106	WOLR790101
RICJ880110	RICJ880111	GUYH850104
RICJ880115	RICJ880116	COWR900101
ROBB760103	ROBB760104	CORJ870103
ROBB760108	ROBB760109	CORJ870108
ROBB760113	ROBB790101	MIYS990105

ROSM880102	ROSM880103	SNEP660104
SNEP660103		

Table S11. The names of the three physicochemical indices for amino acids.

Hydrophobicity	hydrophilicity	mass

Table S12. The names of the two physicochemical indices for amino acids.

Hydrophobicity	hydrophilicity

References

1. Van Der Walt, S., Colbert, S.C. and Varoquaux, G. (2011) The NumPy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, **13**, 22-30.
2. Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W. and Bright, J. (2020) SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature methods*, **17**, 261-272.
3. Hunter, J.D. (2007) Matplotlib: A 2D graphics environment. *Computing In Science & Engineering*, **9**, 90-95.
4. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. *et al.* (2011) Scikit-learn: Machine Learning in Python. *journal of machine learning research*, **12**, 2825-2830.
5. Lemaître, G., Nogueira, F. and Aridas, C.K. (2017) Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *The Journal of Machine Learning Research*, **18**, 559-563.
6. Paszke, A., et al. (2019), *Advances in Neural Information Processing Systems*, Vol. 32, pp. 8026-8037.
7. Altschul, S.F. and Koonin, E.V. (1998) Iterated profile searches with PSI-BLAST - a tool for discovery in protein databases. *Trends Biochem Sci*, **23**, 444-447.
8. Jones, D.T. (1999) Protein secondary structure prediction based on position-specific scoring matrices. *Journal of Molecular Biology*, **292**, 195-202.
9. Cuff, J.A. and Barton, G.J. (2000) Application of multiple sequence alignment profiles to improve protein secondary structure prediction. *Proteins-structure Function & Bioinformatics*, **40**, 502-511.
10. Yang, Y., Heffernan, R., Paliwal, K., Lyons, J., Dehzangi, A., Sharma, A., Wang, J., Sattar, A. and Zhou, Y. (2017) SPIDER2: A Package to Predict Secondary Structure, Accessible Surface Area, and Main-Chain Torsional Angles by Deep Neural Networks. *Methods Mol Biol*, **1484**, 55-63.
11. Lorenz, R., Bernhart, S.H., Honer Zu Siederdissen, C., Tafer, H., Flamm, C., Stadler, P.F. and Hofacker, I.L. (2011) ViennaRNA Package 2.0. *Algorithms Mol Biol*, **6**, 26.
12. Pupko, T., Bell, R.E., Mayrose, I., Glaser, F. and Ben-Tal, N. (2002) Rate4Site: an algorithmic tool

for the identification of functional regions in proteins by surface mapping of evolutionary determinants within their homologues. *Bioinformatics*, **18 Suppl 1**, S71-77.

13. Puntervoll, P., Linding, R., Gemund, C., Chabanis-Davidson, S., Mattingsdal, M., Cameron, S., Martin, D.M., Ausiello, G., Brannetti, B., Costantini, A. *et al.* (2003) ELM server: A new resource for investigating short functional sites in modular eukaryotic proteins. *Nucleic Acids Res*, **31**, 3625-3630.
14. Pugalenthi, G., Suganthan, P.N., Sowdhamini, R. and Chakrabarti, S. (2008) MegaMotifBase: a database of structural motifs in protein families and superfamilies. *Nucleic Acids Res*, **36**, D218-221.
15. Suykens, J.A. and Vandewalle, J.J.N.p.l. (1999) Least squares support vector machine classifiers. **9**, 293-300.
16. Liu, B. (2019) BioSeq-Analysis: a platform for DNA, RNA and protein sequence analysis based on machine learning approaches. *Briefings in Bioinformatics*, **20**, 1280-1294.
17. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R. and Dubourg, V.J.t.J.o.m.L.r. (2011) Scikit-learn: Machine learning in Python. **12**, 2825-2830.
18. Liu, B., Gao, X. and Zhang, H.Y. (2019) BioSeq-Analysis2.0: an updated platform for analyzing DNA, RNA and protein sequences at sequence level and residue level based on machine learning approaches. *Nucleic Acids Res*, **47**, e127-e127.
19. Zeng, M., Zhang, F., Wu, F.-X., Li, Y., Wang, J. and Li, M. (2019) Protein–protein interaction site prediction through combining local and global features with deep neural networks. *Bioinformatics*, **36**, 1114-1120.
20. Hochreiter, S. and Schmidhuber, J. (1997) Long short-term memory. *neural computation*, **9**, 1735-1780.
21. Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y. (2014), *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing* Association for Computational Linguistics, pp. 1724-1734.
22. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I. (2017), *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Curran Associates Inc., Long Beach, California, USA, pp. 6000–6010.
23. Ahmed, K., Keskar, N.S. and Socher, R. (2017) Weighted transformer network for machine translation. *Preprint at https://arxiv.org/abs/1711.02132*.
24. Kitaev, N., Kaiser, Ł. and Levskaya, A. (2020) Reformer: The efficient transformer. *Preprint at https://arxiv.org/abs/2001.04451*.
25. Schmidt, M., Fung, G. and Rosales, R. (2007), *Proceedings of the 18th European conference on Machine Learning*. Springer-Verlag, Warsaw, Poland, pp. 286–297.
26. Bilgic, B., Chatnuntawech, I., Fan, A.P., Setsompop, K., Cauley, S.F., Wald, L.L. and Adalsteinsson, E. (2014) Fast image reconstruction with L2-regularization. *Journal of Magnetic Resonance Imaging*, **40**, 181-191.
27. Frey, B.J. and Dueck, D. (2007) Clustering by passing messages between data points. *Science (New York, N.Y.)*, **315**, 972-976.
28. Ester, M., Kriegel, H.-P., Sander, J. and Xu, X. (1996), *Proceedings of the Second International*

Conference on Knowledge Discovery and Data Mining. AAAI Press, Portland, Oregon, pp. 226–231.

29. Kim, S.C. and Kang, T.J. (2007) Texture classification and segmentation using wavelet packet frame and Gaussian mixture model. *Pattern Recogn.*, **40**, 1207–1221.
30. Skarmeta, A.G., Bensaid, A. and Tazi, N. (2000) Data mining for text categorization with semi‐supervised agglomerative hierarchical clustering. *International journal of intelligent systems*, **15**, 633-646.
31. Jain, A.K., Murty, M.N. and Flynn, P.J. (1999) Data clustering: a review. *ACM computing surveys*, **31**, 264-323.
32. Chandrashekar, G. and Sahin, F. (2014) A survey on feature selection methods. *Computers Electrical Engineering*, **40**, 16-28.
33. Guyon, I. and Elisseeff, A. (2003) An introduction to variable and feature selection. *Journal of machine learning research*, **3**, 1157-1182.
34. Darst, B.F., Malecki, K.C. and Engelman, C.D. (2018) Using recursive feature elimination in random forest to account for correlated variables in high dimensional data. *Bmc Genet*, **19**, 353-363.
35. Sugumaran, V., Muralidharan, V. and Ramachandran, K. (2007) Feature selection using Decision Tree and classification through Proximal Support Vector Machine for fault diagnostics of roller bearing. *Mechanical Systems Signal Processing*, **21**, 930-942.
36. Yeung, K.Y. and Ruzzo, W.L. (2001) Principal component analysis for clustering gene expression data. *Bioinformatics*, **17**, 763-774.
37. Schölkopf, B., Smola, A.J. and Müller, K.-R. (1997), *Proceedings of the 7th International Conference on Artificial Neural Networks*. Springer-Verlag, pp. 583–588.
38. Wei, J.-J., Chang, C.-J., Chou, N.-K. and Jan, G.-J. (2001) ECG data compression using truncated singular value decomposition. *Trans. Info. Tech. Biomed.*, **5**, 290–299.
39. Chang, C.C. and Lin, C.J. (2011) LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, **2**, Article 27.
40. Breiman, L. (2001) Random Forests. *Mach. Learn.*, **45**, 5–32.
41. Sutton, C. and McCallum, A. (2012) An Introduction to Conditional Random Fields. *Found. Trends Mach. Learn.*, **4**, 267–373.
42. Hanson, J., Yang, Y.D., Paliwal, K. and Zhou, Y.Q. (2017) Improving protein disorder prediction by deep bidirectional long short-term memory recurrent neural networks. *Bioinformatics*, **33**, 685-692.
43. Chawla, N.V., Bowyer, K.W., Hall, L.O. and Kegelmeyer, W.P. (2002) SMOTE: synthetic minority over-sampling technique. *journal of artificial intelligence research*, **16**, 321-357.
44. Farquad, M.A.H. and Bose, I. (2012) Preprocessing unbalanced data using support vector machine. *Decision Support Systems*, **53**, 226-233.
45. Junsomboon, N. and Phienthrakul, T. (2017), *Proceedings of the 9th International Conference on Machine Learning and Computing*. Association for Computing Machinery, Singapore, Singapore, pp. 243–247.