**Supplementary material**

**Section A: Hyperparameter tuning**

The selected values for hyperparameter tuning of model by grid search are given as:

|  |  |
| --- | --- |
| **Layers** | **Hyperparameter Settings** |
| Embedding | Output dimension: 41, 64, 100, 128 |
| BiLSTM\_1 | LSTM units: 16, 24, 32, 45, 64 |
| BiLSTM\_2 | LSTM units: 8, 16, 24  Return\_sequences: True, False |

**Section B: Alternative encoding schemes:**

The details of another than embedding encoding are given below:

1. **One-hot encoding:**

A protein sequence of length L having AA amino acids can be represented into a vector of (L x AA) features. In our case, the length of the fragment length is 41, and a total of 20 amino acids. Thus, final dimension of onehot is (41 x 20).

Example: Alanine (A) is represented as 10000000000000000000 and Cysteine (C) is represented as 01000000000000000000 and so on.

1. **PCA encoding (quantitative representation):**

In PCA encoding every amino acid represented by a five-dimensional vector space which derived from multi-dimensional scaling of 23 Physical-chemical properties. In their method used five principal components to reproduce the main variations of 237 properties of natural 20 amino acids which are shown below.

|  |
| --- |
| "A" = [ 0.008, 0.134,-0.475,-0.039, 0.181]  "R" = [ 0.171,-0.361, 0.107,-0.258,-0.364] |
| "N" = [ 0.255, 0.038, 0.117, 0.118,-0.055] |
| "D" = [ 0.303,-0.057,-0.014, 0.225, 0.156] |
| "C" = [-0.132, 0.174, 0.070, 0.565,-0.374] |
| "Q" = [ 0.149,-0.184,-0.030, 0.035,-0.112] |
| "E" = [ 0.221,-0.280,-0.315, 0.157, 0.303] |
| "G" = [ 0.218, 0.562,-0.024, 0.018, 0.106] |
| "H" = [ 0.023,-0.177, 0.041, 0.280,-0.021] |
| "I" = [-0.353, 0.071,-0.088,-0.195,-0.107] |
| "L" = [-0.267, 0.018,-0.265,-0.274, 0.206] |
| "K" = [ 0.243,-0.339,-0.044,-0.325,-0.027] |
| "M" = [-0.239,-0.141,-0.155, 0.321, 0.077] |
| "F" = [-0.329,-0.023, 0.072,-0.002, 0.208] |
| "P" = [ 0.173, 0.286, 0.407,-0.215, 0.384] |
| "S" = [ 0.199, 0.238,-0.015,-0.068,-0.196] |
| "T" = [ 0.068, 0.147,-0.015,-0.132,-0.274] |
| "W" = [-0.296,-0.186, 0.389, 0.083, 0.297] |
| "Y" = [-0.141,-0.057, 0.425,-0.096,-0.091] |
| "V" = [-0.274, 0.136,-0.187,-0.196,-0.299] |
|  |

1. **Protein vector (ProtVec):**

Protvec is a dense representation which is proposed for protein sequences. In this approach, each biological sequence is embedded in an n-dimensional vector that characterizes biophysical and biochemical properties of sequences using the Skip-gram neural network. This vector then oganized after analyzing of biophysical and biochemical properties of this embedding qualitively and quantitively. This 100 dimensional vector constructed by using 3-mer residue of protein sequence. This representation technique is pre-trained, just apply the prediction model. In our case, the length of fragment is 41 so, after applying the 3-mer and protvec the final dimensions become (39,100).

**Section C: Alternative Models:**

The details of the nine different architecture by using a different type of encoding technique. The 5-Fold cross-validation and independent testing AUC is shown.

1. **CNN-onehot model:**

We used one-hot encoding and takes it as input to the CNN based deep learning architecture which details shown as

Input(41,20), Conv1D(8, 3), Dropout (0.1), Conv1D(16, 3), Dropout (0.2), MaxPooling1D(pool\_size=2), Conv1D(24,5), Dropout (0.2), Conv1D(32,7), Dropout (0.4), MaxPooling1D(pool\_size=2), Dense(20), Dropout (0.5), Dense(2).

|  |  |
| --- | --- |
| 5-folds cross-validation ROC | Independent test ROC |

1. **BiLSTM-onehot model:**

We used one-hot encoding and takes it as input to BiLSTM based deep learning architecture which details shown as

Input(41,20), BiLSTM(24), Dropout (0.1), BiLSTM(16), MaxPooling1D(pool\_size=2), Dropout (0.2), Dense(2).

|  |  |
| --- | --- |
| 5-folds cross-validation ROC | Independent test ROC |

1. **CNN-onehot-PCA model:**

we used combined one-hot and PCA vector and takes it as input to the CNN based deep learning architecture which details shown as

Input(41,25), Conv1D(8, 3), Dropout (0.1), Conv1D(16, 3), Dropout (0.2), MaxPooling1D(pool size=2), Conv1D(24,5), Dropout (0.3), Conv1D(32,7), Dropout (0.4), MaxPooling1D(pool size=2), LSTM(32), Dropout (0.4), LSTM(64), Dropout (0.5), Dense(2).

|  |  |  |
| --- | --- | --- |
| 5-folds cross-validation ROC |  | Independent test ROC |

1. **BiLSTM-onehot-PCA**

we used combined one-hot and PCA vector and takes it as input to the BiLSTM based deep learning architecture which details shown as

Input(41,25), BiLSTM(28), Dropout (0.2), BiLSTM(24), MaxPooling1D(pool\_size=2), Dropout (0.3), Dense(2).

|  |  |
| --- | --- |
| 5-folds cross-validation ROC | Independent test ROC |

1. **CNN-emb model:**

we used integer based vector given as input to the embedding layer and the passed to the CNN based layers which details shown as

Input(41,), Embedding(24,64), Conv1D(16, 5), Dropout (0.2), Conv1D(32,7), Dropout (0.3), MaxPooling1D(pool size=2), Dense(24), Dropout (0.4), Dense(1).

|  |  |
| --- | --- |
| 5-folds cross-validation ROC | Independent test ROC |

1. **LSTM-emb model:**

we used integer based vector given as input to the embedding layer and LSTM layers which details shown as

Input(41,), Embedding(24,64), LSTM(64), Dropout (0.2), LSTM(24), Dropout (0.2), Dense(16), Dropout (0.4), Dense(1).

|  |  |
| --- | --- |
| 5-folds cross-validation ROC | Independent test ROC |

1. **CNN-Protvec model:**

We used the prot2vec based encoding technique and given this matrix to CNN based deep learning method which details shown below

Input(39,100), Conv1D(8, 3), Dropout (0.2), MaxPooling1D(pool size=2), Conv1D(16,5), Dropout (0.3), MaxPooling1D(pool size=2), Dense(24), Dropout (0.5), Dense(2).

|  |  |
| --- | --- |
| 5-folds cross-validation ROC | Independent test ROC |

1. **BiLSTM-Protvec model:**

We used the prot2vec based encoding technique and given this matrix to CNN based deep learning method which details shown below

Input(39,100), BiLSTM(64), Dropout (0.1), BiLSTM(32), MaxPooling1D(pool\_size=2), Dropout (0.2), Dense(2).

|  |  |
| --- | --- |
| 5-folds cross-validation ROC | Independent test ROC |

1. **CNN-onehot-emb model:**

we used an integer based vector given as input to the embedding layer and also one-hot encoding which concatenates with the embedding vector and feed forward to further deep learning architecture which details shown as

Input(41,), Input(41,20), Embedding(24,32), Concatenate((41,20),(41,32)), Conv1D(16, 3), Dropout (0.1), Conv1D(24, 5), Dropout (0.3), MaxPooling1D(pool size=2), Conv1D(32, 7), MaxPooling1D(pool size=2), Dense(16), Dropout (0.5), Dense(1).

|  |  |
| --- | --- |
| 5-folds cross-validation ROC | Independent test ROC |