Chapter 6

Artificial Intelligence based Discovery of low hemolytic therapeutic peptides

Therapeutic peptides have gained much attention as an alternative to conventional antibiotics. Although much research has been undertaken to identify peptide-based drugs, few are commercially accessible. The toxicity of peptides is the key roadblock to the development of therapeutic peptides as medications. As a result, identifying low hemolytic peptides is crucial for creating novel peptide-based therapeutics. Wet lab experiments to discover low hemolytic peptides are labor-intensive, time-consuming, and involve testing mammalian red blood cells. As a result, to discover therapeutic peptides with low hemolytic activity, the in-silico tool is needed. Thus, this chapter introduces an artificial intelligence-based framework named EnDL-HemoLyt, which has been made available as a web app at https://endl-hemolyt.anvil.app/. This app will aid wet lab researchers to identify therapeutic peptides with low hemolytic activity.

6.1 Introduction

Therapeutic peptides have gained a lot of attention as an alternative to conventional antibodies and small molecule-based medicines due to their greater tissue penetration, enhanced specificity, minimal resistance, and relatively low manufacturing cost [117, 118, 119, 120]. As a consequence, a lot of studies have been published in recent years detailing the identification of new peptides with therapeutic capabilities ranging from antimicrobial, antibacterial, antifungal, antiviral, antimalarial, antiparasitic, anticancer, cell-penetrating, tumor-homing, antihypertensive, etc. Although much research has been undertaken to identify peptide-based drugs, only a few are commercially accessible. The toxicity of peptides is the key roadblock to the development of therapeutic peptides as medications. It can be determined by examining the integrity of the cell membrane, the viability of the cell, and the cytotoxic effects. Numerous laboratory-based techniques for determining peptide toxicity are available, including ATP-based assay [121], lactate dehydrogenase leakage assay [122], 3-(4,5-dimethyl-2-thiazolyl)-2, 5-diphenyl-2H-tetrazolium bromide assay [123], and hemolytic assay [124]. Among these methods, the hemolytic activity of peptides against red blood cells (RBCs) is regarded as the first-line technique [125]. Hemolysis is a condition when RBCs are destroyed before reaching their anticipated lifetime of 120 days. The situation becomes more severe when the destruction of RBCs exceeds its creation due to hemolysis. As a result, the hemoglobin level and oxygen-carrying capability of blood decrease, leading to anemia, the most prevalent kind of blood illness that impacts millions of individuals globally. Therefore, high hemolytic peptides are not suitable in pharmaceutical formulations. As a result, identifying low hemolytic peptides is crucial for creating novel peptide-based therapeutics. Experiments to discover low hemolytic peptides are labor-intensive, time-consuming, and involve testing mammalian red blood cells.

Thus, wet-lab researchers often perform *in-silico* prediction to select low hemolytic peptides before proceeding with *in-vitro* testing. HLPpred-Fuse [3], HemoPred [4],

and HemoPI [5] are some of the *in-silico* tools that can classify peptides as **h**igh-h**e**molytic **p**eptide**s** (HEPs) and **l**ow-h**e**molytic **p**eptide**s** (LEPs) and therefore can serve the purpose. But the aforementioned tools have some limitations: (i) **N/C terminal modifications:** DBAASP v3 [126] is the most recent database detailing the hemolytic data of therapeutic peptides. Over half of the therapeutic peptides in this database have either N-terminal acetylation (NTC) or C-terminal amidation (CTM), or both. Existing tools are designed without accounting for such N/C terminal modifications , and therefore they can't make predictions for peptides with such modifications. (ii) **Amount of data:** Data is food for Artificial intelligence (AI). As a result, there has been a recent push in the AI community toward data-centric AI (improving data to enhance performance) from model-centric AI (improving the algorithm to enhance performance). The data-centric AI competition, which was recently launched, embodies this trend toward data engineering. Using sample problems, it was shown that the data-centric method outperformed the baseline by a large margin, but the model-centric approach failed to do so [24, 25, 26, 27]. From the preceding discussion, it is evident that data is critical in providing an AI-based solution to a problem. However, the dataset associated with the current tools is small and does not include the therapeutic peptides found in the last eight years , which limits their applicability for wet-lab researchers. (iii) **Performance:** The performance of available tools is also low , which can be improved by enhancing data quality and using better algorithms.

Keeping the aforementioned limitations in mind, current work has (i) Utilised the recent dataset (that contains standard peptides and the peptides having NTC or CTM or both) from the DBAASP v3 database. The experimentally validated data in the DBAASP v3 database is provided as the percentage of hemolysis induced by the specified peptide concentration. To translate this data into the desired information (HEPs/LEPs), well-established criteria given in Table 6.1 has been utilized. (ii) Developed an ensemble classifier that can classify a peptide as HEP or LEP. For building

the ensemble classifier, the decisions provided by bidirectional long short-term memory (BiLSTM), bidirectional temporal convolutional network (BiTCN), and 1-dimensional convolutional neural network (1DCNN) deep learning algorithms are combined using a min/max combiner technique. Though handcrafted features (HCF) are not mandatory for deep learning algorithms [103], but still HCF were externally supplied to deep learning algorithms so that they can learn the features that are absent from HCF, and a better feature vector can be constructed by concatenating HCF and deep learning-based features (DLF). HLPpred-Fuse, HemoPred, and HemoPI have utilized HCF corresponding to the 20 standard amino acids with numerous machine-learning algorithms. The peptides having NTC or CTM or both are considered in the current study; therefore, the HCF obtained from 20 standard amino acids are not applicable. Thus to compute HCF, the given peptide sequences are first converted into smiles, then NTC and CTM are performed, and then Mordred [127] is used to generate molecular descriptors.

The graphical abstract of the proposed work is shown in Figure 6.1, which contains four major components: (i) **N-terminal acetylation / C-terminal amidation:** In the present work, both HCF and DLF are used. HCF are obtained by calculating molecular descriptors from smiles strings of the peptide, whereas DLF were computed by deep learning algorithms. The standard peptides (peptides made of 20 standard amino acids) and peptides with modifications (NTC or CTM or both) have been considered. To include these modifications, the smile strings based on standard peptides have been calculated, and then after making the required modifications, the molecular descriptors are computed. Moreover, to incorporate these modifications into DLF, the standard amino acids are substituted by others. After prepossessing, the peptides are fed to the deep-learning models. (ii) **BiLSTM / BiTCN / 1DCNN deep learning algorithms:** To classify peptides as HEPs/LEPs, three heterogeneous deep learning algorithms: BiLSTM, BiTCN, and 1DCNN have been utilized. (iii) **Ensemble Classifier:** The aforementioned deep learning algorithms are heterogeneous. Therefore, they

have been combined using a min/max combiner ensemble algorithm to provide better results. Other than the min/max combiner algorithm, several ensemble approaches have also been investigated. The min/max combiner approach performed better and therefore has been considered over others. (iv) Based on the decision provided by the ensemble classifier, the peptide is classified as HEP/LEP.

Ablation studies have also been performed to understand the contribution of the ensemble algorithm, HCF, and DLF in the proposed framework. From the ablation studies, it was found that all of these contributed to the performance of the proposed model.

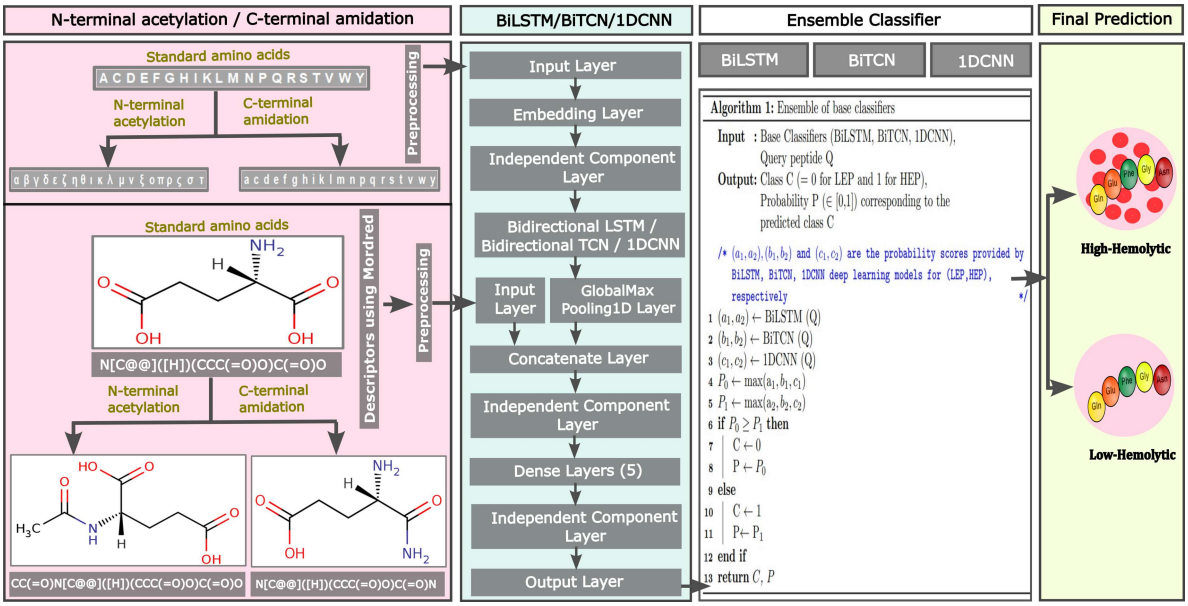

Figure 6.1: Proposed Framework.

The main contributions of our chapter are summarised as follows:

1. In the current work, a novel framework has been proposed, which integrates deep learning algorithms, namely BiLSTM, BiTCN, and 1DCNN, using the min/max combiner ensemble technique and makes use of features from both DLF and HCF.
2. The proposed framework is trained and tested on the recent dataset, which contains standard peptides and peptides with either N-terminal acetylation or C-terminal amidation, or both.

3. Ablation studies have also been performed to understand the contribution of the ensemble algorithm, HCF, and DLF in the proposed framework.
4. To help wet-lab researchers identify low hemolytic therapeutic peptides, the model developed from the proposed framework has been set up as a web server and is freely accessible online at https://endl-hemolyt.anvil.app/.
5. The web server can classify low and high hemolytic peptides better than existing tools and can also provide predictions for N/C terminal-modified peptides. Additionally, it can undertake mutation analysis and residue scans to select the optimum amino acid at each position in a peptide sequence.

The rest of this chapter is structured as follows. The information regarding the dataset and overall idea of the proposed work is presented in Section 6.2. The results are presented in Section 6.3. The information related to web server is presented in Section 6.4. The conclusion and future work are provided in Section 6.5.

6.2 Materials and methods

6.2.1 Dataset

The dataset D_s comprised 4,339 peptides. DBAASP v3 database has been utilized to prepare D_s. After acquiring the peptide sequences from the DBAASP v3 database, the D_s was generated using the following steps: (i) The peptides with a length $\in$ [5,50] for which hemolytic data was available were selected. (ii) The peptides comprising natural amino acids with no N/C terminal modifications or any of the modifications (NTC or CTM or both) were selected. (iii) For some of the peptides, the hemolytic concentration is available in μM, whereas, for others, it is available in $\mu g/ml$. Therefore hemolytic concentration for all the peptides was also converted to μM. (iv) For labeling a peptide as HEP or LEP, the well-established criteria outlined in Table 6.1 from the previous study [3, 4, 5] has been used. (v) Duplicate peptides, those labeled as both HEPs and

LEPs (due to distinct experimentation conditions, several peptides had multiple entries and therefore got labeled as both HEPs and LEPs), and those not labeled at all were eliminated. (vi) To further refine the dataset, CD-HIT2D with a cutoff value of 0.9 has been used.

Table 6.1: Criteria for classifying a peptide as **h**igh-h**e**molytic **p**eptide (HEP) and **l**ow-h**e**molytic **p**eptide (LEP) [3, 4, 5]

HEP		**LEP**	
Activity (%)	Concentration (μM)	Activity (%)	Concentration (μM)
≥ 5	≤ 10	≤ 2	≥ 10
≥ 10	≤ 20	≤ 5	≥ 20
≥ 15	≤ 50	≤ 10	≥ 50
≥ 20	≤ 100	≤ 15	≥ 100
≥ 30	≤ 200	≤ 20	≥ 200
≥ 50	≤ 300	≤ 30	≥ 300
MHC	≤ 50	≤ 50	≥ 500
		MHC	≥ 100

6.2.2 Proposed Framework

In the current work, a novel framework has been proposed. The peptide is a sequence of amino acids. Deep learning algorithms, namely BiLSTM, BiTCN, and 1DCNN, have been extensively used in sequence-related tasks in the field of Natural language processing and bioinformatics [75, 128, 71, 129, 76, 130, 111, 131, 77, 112, 78]. Therefore, in the current work, BiLSTM, BiTCN, and 1DCNN deep learning algorithms have been utilized. In the proposed framework, the decisions of the aforementioned deep learning algorithms have been combined using a mix/max combiner algorithm.

Deep learning algorithms do not require domain expertise to extract features (DLF)

from the data. However, instead of relying exclusively on DLF, HCF are employed. The goal of employing HCF is to train deep-learning algorithms to learn features that HCF lacks so that a superior feature vector can be created by concatenating HCF and DLF. The details of the proposed framework are provided in this Section. For BiLSTM, BiTCN, and 1DCNN deep learning algorithms:

The first layer is the Input layer, using which the data (peptides) has been provided. Before feeding the peptides, three significant pre-processing steps were performed: (i) Modified the standard peptides (peptides made of 20 standard amino acids) to account for NTC and CTM. For NTC and CTM, the standard amino acids (A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y) are substituted by $(\alpha, \beta, \gamma, \delta, \epsilon, \zeta, \eta, \theta, \iota, \kappa, \lambda, \mu, \nu, \xi, o, \pi, \rho, \sigma, \tau, \upsilon)$ and (a, c, d, e, f, g, h, i, k, l, m, n, p, q, r, s, t, v, w, y), respectively. (ii) Mapped 60 amino acids (20 standard + 20 NTC + 20 CTM) of peptides to numerical values (1, 2,..., 60). (iii) Made all peptides of uniform length 50 (= maximum length of peptide). Therefore, the shape of output from this layer is (b_s, 50), where b_s stands for batch size (= 16 in our case).

After the Input layer,the Embedding layer is used. The embedding layer basically learns the vector representation for every amino acid present in the dictionary. Like other layers Embedding layer can be initialized with random weights or pre-trained weights (embeddings), which can help this layer learn the vector representation in a better way. In the current work, the word2vec algorithm is used to learn the pre-trained weights for this layer. The length of vector representation learned by word2vec for each amino acid is 100. Therefore, the shape of output from this layer is (b_s, 50, 100).

After the Embedding layer, an Independent component layer (ICL) is used. The ICL was presented for the first time in [46], where the authors merged batch normalization and dropout. They ran multiple experiments and observed that ICL leads to a stable training process, quicker convergence, and improved generalization. For preparing ICL,

batch normalization and a one-dimensional spatial dropout layer (which performs the same function as Dropout, however, it drops entire 1D feature maps instead of individual elements) were included. This layer does not make any change in the shape of the input. Therefore, the shape of output from this layer is (b_s, 50, 100).

Next, corresponding to BiLSTM, BiTCN, and 1DCNN algorithms, BiLSTM, BiTCN, and 1DCNN layers, respectively, have been added. In BiLSTM, the number of neurons used with each LSTM layer is 128. In BiTCN, the number of filters used with each TCN layer is 128. In 1DCNN number of filters used is 256. Therefore, the shape of output from this layer is (b_s, 50, 256).

After this, the GlobalMaxPooling1D layer is used to perform downsampling of input by taking the maximum value over the time dimension. The shape of output from this layer is (b_s, 256).

After the GlobalMaxPooling1D layer, the Concatenate layer is used to fuse the vector from the GlobalMaxPooling1D layer and HCF from Mordred (provided to the proposed framework using Input layer). To compute HCF, the given peptide sequences are first converted into smiles, then the NTC and CTM (if any) are performed, and finally, Mordred is used to generate different molecular descriptors. The length of HCF computed by Modered is 1826. Therefore the shape of output from this layer is (b_s, 256 + 1826).

After Concatenate layer, ICL (batch normalization + regular dropout layer) is used. This layer does not make any change in the shape of the input. Therefore, the shape of output from this layer is (b_s, 2082).

The output from the preceding ICL is fed to dense layers (D_1, D_2, D_3, D_4, D_5). The number of neurons used with (D_1, D_2, D_3, D_4, D_5) was (1024, 256, 128, 64, 8), respectively. The shape of output from (D_1 , D_2, D_3, D_4, D_5) is (b_s, 1024), (b_s, 256), (b_s, 128), (b_s, 64) and (b_s, 8), respectively. Rectified linear unit (ReLU) activation function is used with BiLSTM, BiTCN, 1DCNN, and dense layers

After D_5, the ICL (batch normalization + regular dropout layer) was used. This layer does not make any change in the shape of the input. Therefore, the shape of output from this layer is (b_s, 8).

After ICL, an output layer of two neurons was incorporated. The shape of output from this layer is (b_s, 2). Corresponding to each peptide, it provides the probability of HEP and LEP. If Probability(LEP)$\geq$ Probability(HEP), then the peptide is predicted as LEP; else predicted as HEP.

The activation function used is softmax (σ).

The loss function(L) used is categorical cross-entropy, which can be defined as follows:

$$L = -\sum_{i=1}^{2} y_i \log(p_i) \tag{6.1}$$

The network weights were updated using the Adam (Adaptive Moment Estimation) optimizer.

Finally, the predictions obtained from the base classifiers BiLSTM, BiTCN, and 1DCNN were integrated to determine the class of the query peptide (described in Section 6.2.2.1).

6.2.2.1 Ensemble of base classifiers

The algorithms BiLSTM, BiTCN, and 1DCNN, are heterogeneous in nature. As a result, peptides that got misclassified by one or the other classifier might be correctly classified by remaining. Therefore, the performance can be improved by integrating the predictions made by these classifiers. Wet-lab researchers use the *in-silico* approach to reduce thousands of candidate peptides to a handful for wet-lab synthesis and investigation. Therefore, even little improvements in the tool's performance may significantly save their time and money. In literature, researchers combined the predictions from the deep learning algorithms using various fuzzy-based approaches like authors in [132]

performed a fuzzy rank-based fusion of classifiers by considering two non-linear functions on the decision scores generated by base learners, authors in [133] performed a fuzzy rank-based fusion of CNN models using Gompertz function and authors in [134] performed a fuzzy distance-based ensemble of deep models. Aside from the aforementioned fuzzy-based approaches, there are computationally effective ensemble strategies such as min/max combiner, average combiner, and majority voting, the performance of which is determined to be significantly better than the individual models by the authors in [135].

6.3 Experiments and Results

This section briefly describes the experimental configuration, performance metrics, assessment procedure used, results obtained from the proposed framework. We have also conducted the ablation studies to understand the contribution of different components in the proposed framework. Moreover, we have also performed additional experiments using HCF. This section also provides the results obtained from these ablation studies and additional experiments.

6.3.1 Experimental Configuration

The deep learning algorithms were implemented using Keras deep learning library [48] with Tensorflow as the backend, and machine learning algorithms were implemented using scikit-learn [49]. All experiments were carried out on a CPU compute node configured with a 2.4 GHz Intel-Xeon Skylake 6148 processor and 192 GB RAM.

6.3.2 Performance Metrics

To access the performance, we used accuracy (A_{cc}), recall/sensitivity (S_n), precision (P_r), F1-Score (F_s), specificity (S_p), balanced accuracy (B_a), Matthews correlation coefficient (MCC).

6.3.3 Assessment Procedure

The D_s has been divided into five folds: Fold 0,..., Fold 4. Using the aforementioned five-folds, five splits were constructed: Split 0,..., and Split 4. In each Split s ($0 \leq s \leq 4$), test data (S^{Test}) was obtained from Fold s, validation data (S^{Val}) was obtained from Fold (s +1) mod 5, and training data (S^{Train}) was obtained from the remaining three folds. This process ensures that each fold is utilized exactly once for validation and testing. Moreover, $S^{Train} \cap S^{Test} = \phi$, $S^{Train} \cap S^{Val} = \phi$, and $S^{Test} \cap S^{Val} = \phi$, which ensures that there is no problem of peeking [136, 137]. Let P0,..., P4 be the performance metrics derived from Split 0,..., Split 4, respectively. Performance (P_{re}) is calculated as Mean (P_μ) $\pm$ Standard deviation (P_σ).

$$
\begin{aligned}
P_\mu &= \frac{\sum_{i=0}^{i=4} P_i}{5} \\
P_\sigma &= \sqrt{\frac{\sum_{i=0}^{i=4} (P_i - P_\mu)^2}{5}} \\
P_{re} &= P_\mu \pm P_\sigma
\end{aligned}
\tag{6.2}
$$

As described in Section 6.2.2.1, several ensemble algorithms can be used to combine the base classifiers. Therefore, S^{Val} from five folds was utilized to identify the best-performing algorithm among these. Table 6.2 display the results obtained by the individual and ensemble models. The comparison between these results is shown in Fig 6.2. As can be seen from this figure, ensemble models performed well compared to individual models, and the best performance is obtained with the min/max combiner ensemble algorithm. Therefore, the min/max combiner ensemble algorithm has been considered in the current work.

Table 6.2: Results obtained by BiTCN, BiLSTM, 1DCNN, various ensemble techniques utilising HCF and DLF

S.No.	Ensemble Algorithm	Feature	Model	$A_{cc}(\%)$	$S_n(\%)$	$P_r(\%)$	$F_s(\%)$	$S_p(\%)$	$B_a(\%)$	$Mcc(\times 100)$
1	BiLSTM		BiLSTM-HD	84.25 ± 1.03	83.93 ± 2.34	82.41 ± 0.40	83.15 ± 1.32	84.53 ± 0.28	84.23 ± 1.12	68.41 ± 2.16
2	BiTCN		BiTCN-HD	85.61 ± 1.33	84.43 ± 1.41	84.59 ± 2.40	84.48 ± 1.29	86.64 ± 2.50	85.53 ± 1.27	71.12 ± 2.64
3	1DCNN		1DCNN-HD	85.15 ± 0.99	83.49 ± 2.10	84.34 ± 1.16	83.89 ± 1.16	86.59 ± 1.27	85.04 ± 1.03	70.16 ± 2.02
4	Min/Max combiner [135]		MCE-HD	86.74 ± 0.49	85.47 ± 1.28	85.88 ± 1.01	85.66 ± 0.56	87.84 ± 1.10	86.66 ± 0.50	73.36 ± 1.0
5	Average combiner [135]	HCF	ACE-HD	86.26 ± 0.98	85.03 ± 1.22	85.29 ± 1.27	85.15 ± 1.06	87.32 ± 1.20	86.18 ± 0.98	72.38 ± 1.99
6	Majority Voting [135]	+	MVE-HD	86.05 ± 1.21	84.83 ± 1.34	85.06 ± 1.58	84.94 ± 1.28	87.11 ± 1.47	85.97 ± 1.20	71.96 ± 2.43
7	Fuzzy non linear [132]	DLF	FNE-HD	86.26 ± 1.04	84.98 ± 1.30	85.33 ± 1.33	85.15 ± 1.13	87.37 ± 1.24	86.17 ± 1.05	72.38 ± 2.11
8	Fuzzy gompertz [133]		FGE-HD	86.26 ± 0.98	85.03 ± 1.22	85.29 ± 1.27	85.15 ± 1.06	87.32 ± 1.20	86.18 ± 0.98	72.38 ± 1.99
9	Fuzzy distance [134]		FDE-HD	86.26 ± 0.98	85.03 ± 1.22	85.29 ± 1.27	85.15 ± 1.06	87.32 ± 1.20	86.18 ± 0.98	72.38 ± 1.99

Table 6.3: Results obtained by Min/Max combiner algorithm utilizing HCF and DLF

S.No.	Algorithm	Feature	Model	$A_{cc}(\%)$	$S_n(\%)$	$P_r(\%)$	$F_s(\%)$	$S_p(\%)$	$B_a(\%)$	$Mcc(\times 100)$
1	Min/Max combiner	HCF + DLF	MCE-HD	86.65 ± 1.14	85.28 ± 1.74	85.85 ± 1.38	85.55 ± 1.24	87.84 ± 1.35	86.56 ± 1.15	73.17 ± 2.29

Table 6.4: Results obtained by BiTCN, BiLSTM, 1DCNN algorithm utilizing HCF and DLF

S.No.	Algorithm	Feature	Model	$A_{cc}(\%)$	$S_n(\%)$	$P_r(\%)$	$F_s(\%)$	$S_p(\%)$	$B_a(\%)$	$Mcc(\times 100)$
1	BiLSTM	HCF + DLF	BiLSTM-HD	83.22 ± 1.27	82.14 ± 1.91	81.74 ± 1.33	81.93 ± 1.41	84.14 ± 1.27	83.14 ± 1.30	66.28 ± 2.58
2	BiTCN		BiTCN-HD	85.15 ± 0.43	84.03 ± 1.16	83.98 ± 1.26	83.99 ± 0.35	86.12 ± 1.48	85.08 ± 0.38	70.18 ± 0.80
3	1DCNN		1DCNN-HD	85.18 ± 1.04	83.29 ± 1.48	84.52 ± 1.24	83.89 ± 1.14	86.81 ± 1.20	85.05 ± 1.05	70.18 ± 2.11

Table 6.5: Results obtained by BiTCN, BiLSTM, 1DCNN and Min/Max combiner algorithm utilising DLF

S.No.	Algorithm	Feature	Model	$A_{cc}(\%)$	$S_n(\%)$	$P_r(\%)$	$F_s(\%)$	$S_p(\%)$	$B_a(\%)$	$Mcc(\times 100)$
1	BiLSTM	DLF	BiLSTM-DL	82.94 ± 0.82	81.30 ± 1.07	81.82 ± 1.48	81.54 ± 0.79	84.36 ± 1.62	82.83 ± 0.78	65.71 ± 1.62
2	BiTCN		BiTCN-DL	84.62 ± 1.14	83.24 ± 2.29	83.53 ± 0.96	83.37 ± 1.37	85.82 ± 0.94	84.53 ± 1.20	69.10 ± 2.33
3	1DCNN		1DCNN-DL	83.26 ± 1.20	80.20 ± 1.03	83.12 ± 1.69	81.63 ± 1.25	85.91 ± 1.58	83.05 ± 1.18	66.31 ± 2.43
4	Min/Max combiner [135]		MCE-DL	85.50 ± 0.77	83.83 ± 1.25	84.74 ± 1.27	84.27 ± 0.81	86.94 ± 1.32	85.39 ± 0.76	70.85 ± 1.55

6.3.4 Results obtained by the proposed Framework

We have proposed a framework that utilizes transfer learning and ensemble learning techniques with deep learning algorithms. The results obtained by proposed framework on S^{Test} are given in Table 6.3. As can be seen from this table, our proposed framework obtained A_{cc}, S_n, P_r, F_s, S_p, B_a and Mcc values $\approx$ 87, 85, 86, 86, 88, 87, 73 , respectively which is similar to the results obtained for validation data (Table 6.2 S.No.4). These equivalent results obtained indicate that the proposed framework is stable and robust. We also performed ablation studies and conducted additional experiments considering state-of-the-art methods. These ablation studies and additional experiments are evaluated on the test data, and their findings are presented in the subsequent Sections.

6.3.5 Ablation Studies

To understand the contribution of (i) Ensemble algorithm, (ii) Handcrafted features (HCF) (iii) Deep learning-based features (DLF) in the proposed framework, ablation studies have been performed utilizing S^{Test} from five folds.

6.3.5.1 Impact of using Ensemble algorithm

To understand the contribution of the ensemble algorithm, it has been eliminated from the proposed framework. Table 6.3 displays the results obtained for the proposed framework, Table 6.4 displays the results obtained for the individual models, and Figure 6.3 compare their results. As seen in this Figure, A_{cc}, F_s, B_a and Mcc values decreased by $\approx$ 1.5-3.5, 1.5-3.5, 1.5-3.5, 3-7 % when the ensemble algorithm is removed. Specifically, there is a large decrease in the MCC value, which is considered as the best performance metric in evaluating binary classification task [138]. This shows the importance of the Ensemble algorithm in the proposed framework.

The aggregate prediction is always less noisy than the individual prediction, which

can be the probable reason for the better performance of the ensemble algorithm-based model compared to individual models.

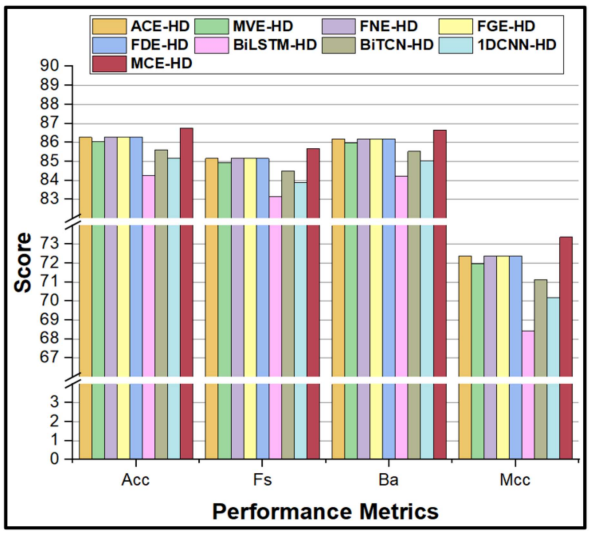

Figure 6.2: Comparison of results obtained on applying both HCF and DLF with (i) average combiner (ACE-HD), (ii) majority voting (MVE-HD), (iii) fuzzy non-linear (FNE-HD), (iv) fuzzy gompertz (FGE-HD), (v) fuzzy distance (FDE-HD), (vi) BiLSTM (BiLSTM-HD), (vii) BiTCN (BiTCN-HD), (viii) 1DCNN (1DCNN-HD), and (ix) Min/Max combiner (MCE-HD).

6.3.5.2 Impact of using HCF

Despite the fact that HCF are not mandatory for deep learning approaches, HCF have been with BiLSTM, BiTCN, and 1DCNN algorithms. The aim of using HCF is to make deep learning algorithms to learn the features that are missing from HCF so that a better feature vector can be constructed by concatenating HCF and DLF. Therefore, to understand the contribution of HCF, they were removed from the proposed framework. The results attained by the BiLSTM, BiTCN, 1DCNN, and ensemble algorithms after the elimination of HCF are shown in Table 6.5. The results obtained by BiLSTM, BiTCN, 1DCNN, and ensemble classifier when both HCF and DLF were used and when only DLF were utilized are compared in Figure 6.4, Figure 6.5, Figure 6.6 and Figure 6.7 respectively. As shown in these Figures, the removal of HCF has a negative

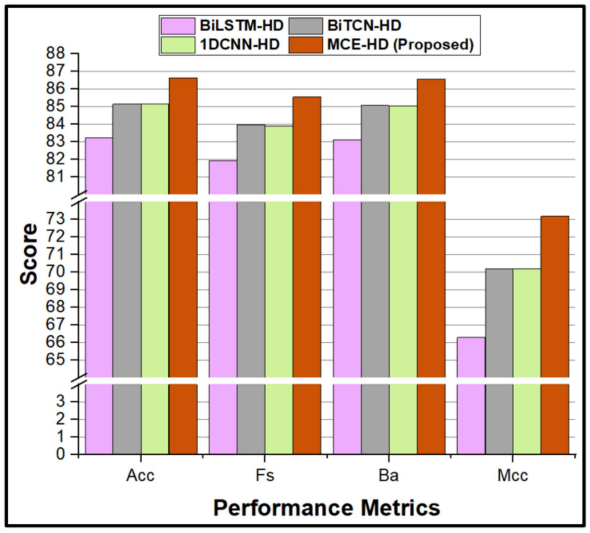

Figure 6.3: Comparison of results obtained on applying both HCF and DLF with (i) BiLSTM (BiLSTM-HD), (ii) BiTCN (BiTCN-HD), (iii) 1DCNN (1DCNN-HD), and (iv) Min/Max combiner (MCE-HD).

impact on the performance of all models. Utilizing an ensemble technique improves performance. However, it is still less than the proposed framework, as shown in Figure 6.7. Specifically, (i) For BiLSTM A_{cc}, F_s, B_a and Mcc values decreased by $\approx$ 0.5, 0.5, 0.5 and 0.5 % when the HCF were eliminated. (ii) For BiTCN A_{cc}, F_s, B_a and Mcc values decreased by $\approx$ 0.5, 0.5, 0.5 and 1 % when the HCF were eliminated. (iii) For 1DCNN A_{cc}, F_s, B_a and Mcc values decreased by $\approx$ 2, 2.5, 2 and 4 % when HCF were eliminated. (iv) For ensemble classifier A_{cc}, F_s, B_a and Mcc values decreased by $\approx$ 1, 1.5, 1 and 2.5 % when the HCF were eliminated.

Deep learning algorithms have reduced our reliance on HCF. However, the importance of HCF cannot be disregarded, as it is not always possible for deep learning algorithms to learn the optimal features required for classification. This can be likely the cause of decreased performance on the removal of HCF.

6.3.5.3 Impact of using DLF

BiLSTM, BiTCN, and 1DCNN deep learning algorithms basically consist of two modules, namely the feature extraction module and classification module (CM). The feature

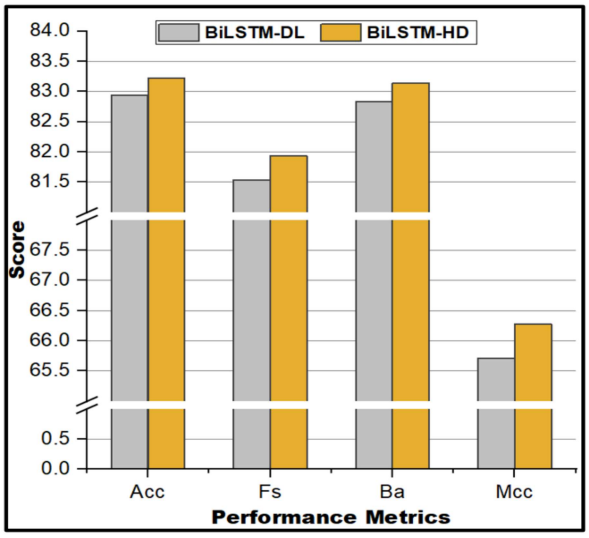

Figure 6.4: Comparison of results obtained on using only DLF with BiLSTM (BiLSTM-DL) and on applying both HCF and DLF with BiLSTM (BiLSTM-HD).

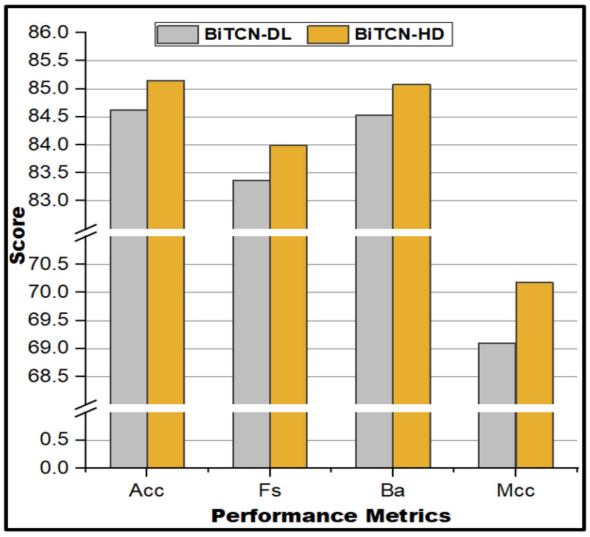

Figure 6.5: Comparison of results obtained on using only DLF with BiTCN (BiTCN-DL) and on applying both HCF and DLF with BiTCN (BiTCN-HD).

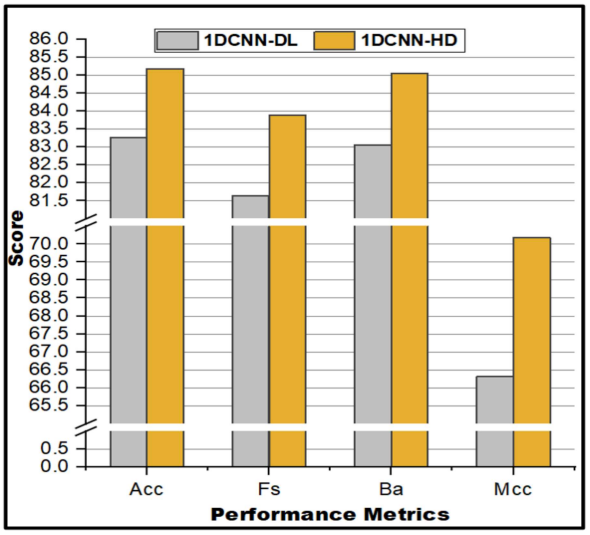

Figure 6.6: Comparison of results obtained on using only DLF with 1DCNN (1DCNN-DL) and on applying both HCF and DLF with 1DCNN (1DCNN-HD).

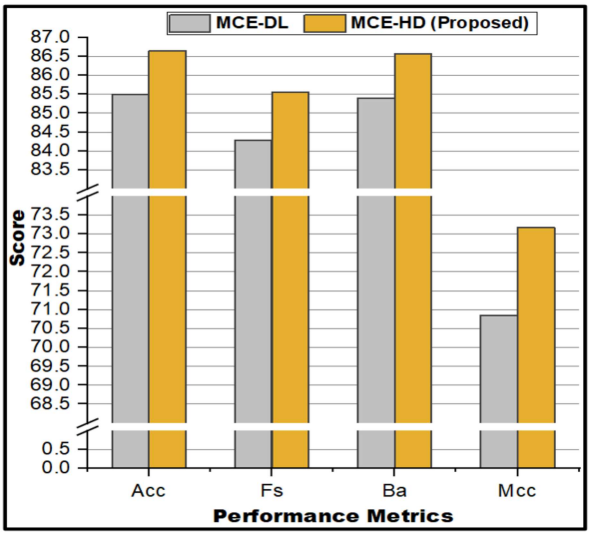

Figure 6.7: Comparison of results obtained on using only DLF with min/max combiner (MCE-DL) and on applying both HCF and DLF with min/max combiner (MCE-HD).

extraction module is responsible for extracting DLF, whereas the classification module is responsible for the classification of DLF. The feature extraction module has been removed, and now the remaining classification module is trained to utilize HCF. The results obtained are shown in Table 6.6, and a comparison with the proposed method is shown in Figure 6.8. As can be seen from this Figure, on removing DLF, there is a heavy loss in performance across all the metrics. Specifically, A_{cc}, F_s, B_a, and Mcc values decreased by $\approx$ 6, 6, 5.5, and 11 % when the DLF were eliminated.

Before the emergence of deep learning algorithms, optimal feature construction relied solely on HCF prepared by domain experts. The peptides can have a wide range of properties. Therefore, it is difficult to have HCF that encodes all the properties of peptides. Deep learning algorithms can automatically extract features from data and, therefore, can learn some complex properties which are not available in the form of HCF. This can be likely the cause of decreased performance on the removal of DLF.

Table 6.6: Results obtained by classification module utilising HCF

S.No.	Algorithm	Feature	Model	$A_{cc}(\%)$	$S_n(\%)$	$P_r(\%)$	$F_s(\%)$	$S_p(\%)$	$B_a(\%)$	$Mcc(\times 100)$
1	CM	HCF	CM-HC	81.10 ± 1.02	79.92 ± 2.44	79.48 ± 1.55	79.66 ± 1.21	82.11 ± 1.95	81.02 ± 1.05	62.06 ± 2.04

Table 6.7: Results obtained by utilizing different machine learning algorithms with different molecular descriptors

S.No.	Model	$A_{cc}(\%)$	$S_n(\%)$	$P_r(\%)$	$F_s(\%)$	$S_p(\%)$	$B_a(\%)$	$Mcc(\times 100)$
1	SVM	77.737 ± 0.84	78.21 ± 1.45	74.92 ± 1.63	76.51 ± 0.70	77.32 ± 2.21	77.77 ± 0.76	55.45 ± 1.57
2	RF	81.49 ± 0.85	77.57 ± 1.21	81.64 ± 1.78	79.53 ± 0.77	84.88 ± 1.92	81.22 ± 0.79	62.76 ± 1.71
3	NB	68.21 ± 1.79	73.74 ± 4.56	63.68 ± 2.25	68.22 ± 1.91	63.44 ± 4.95	68.59 ± 1.72	37.33 ± 3.37
4	ERT	82.69 ± 1.46	79.61 ± 1.45	82.54 ± 2.72	81.01 ± 1.34	85.35 ± 2.89	82.48 ± 1.37	65.20 ± 2.91
5	KNN	78.75 ± 2.81	82.09 ± 1.32	74.78 ± 3.75	78.22 ± 2.32	75.86 ± 4.95	78.98 ± 2.65	57.88 ± 5.17
6	AB	71.83 ± 1.72	69.56 ± 2.15	69.70 ± 2.40	69.60 ± 1.71	73.79 ± 2.84	71.68 ± 1.67	43.40 ± 3.43
7	LR	71.69 ± 2.30	71.95 ± 3.13	68.56 ± 2.46	70.19 ± 2.48	71.47 ± 2.66	71.71 ± 2.32	43.35 ± 4.64
8	GB	78.58 ± 0.88	77.52 ± 1.60	76.60 ± 1.45	77.04 ± 0.91	79.51 ± 1.81	78.51 ± 0.86	57.01 ± 1.75

Table 6.8: Results obtained by utilizing Meta machine learning algorithms with different molecular descriptors

S.No.	Model	$A_{cc}(\%)$	$S_n(\%)$	$P_r(\%)$	$F_s(\%)$	$S_p(\%)$	$B_a(\%)$	$Mcc(\times 100)$
1	Meta-SVM	82.16 ± 0.91	82.15 ± 2.09	80.00 ± 1.99	81.01 ± 0.88	82.17 ± 2.42	82.16 ± 0.86	64.28 ± 1.77
2	Meta-RF	82.80 ± 1.08	81.35 ± 1.62	81.58 ± 2.07	81.43 ± 1.03	84.06 ± 2.34	82.70 ± 1.03	65.46 ± 2.12
3	Meta-NB	82.59 ± 1.38	80.60 ± 2.04	81.72 ± 2.60	81.11 ± 1.33	84.32 ± 2.85	82.46 ± 1.31	65.05 ± 2.74
4	Meta-ERT	82.18 ± 1.11	81.99 ± 2.22	80.19 ± 2.76	81.02 ± 0.88	82.34 ± 3.32	82.17 ± 0.99	64.35 ± 2.13
5	Meta-KNN	82.27 ± 0.85	82.44 ± 2.26	80.03 ± 2.12	81.17 ± 0.79	82.13 ± 2.65	82.29 ± 0.79	64.54 ± 1.62
6	Meta-AB	81.35 ± 1.91	80.65 ± 1.52	79.55 ± 3.12	80.06 ± 1.73	81.95 ± 3.42	81.30 ± 1.82	62.61 ± 3.76
7	Meta-LR	81.95 ± 1.28	81.85 ± 1.06	79.79 ± 2.02	80.79 ± 1.19	82.04 ± 2.23	81.94 ± 1.22	63.82 ± 2.49
8	Meta-GB	82.76 ± 0.65	77.77 ± 3.04	84.06 ± 2.99	80.69 ± 0.67	87.07 ± 3.22	82.42 ± 0.57	65.45 ± 1.45

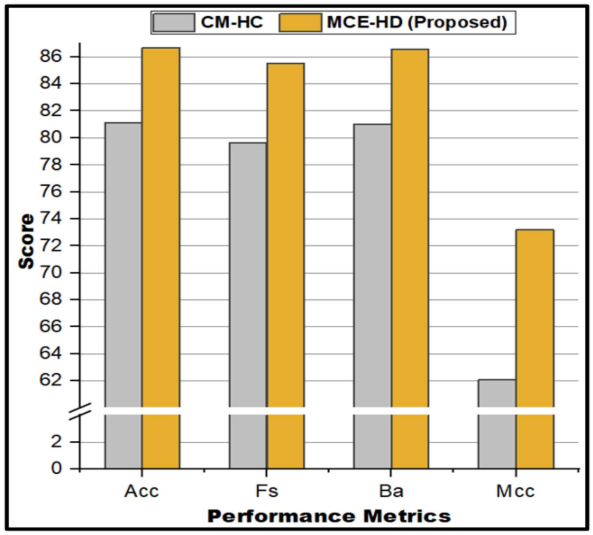

Figure 6.8: Comparison of results obtained on using HCF with classification module (CM-HC) and on applying both HCF and DLF with min/max combiner (MCE-HD).

6.3.6 Additional Experiments and Analysis

6.3.6.1 Utilising the HCF with the machine learning

Additional experiments include utilizing the HCF with the machine learning algorithms (support vector machine (SVM), random forest (RF), naive Bayes (NB), extremely randomized trees (ERT), k-nearest neighbors (KNN), AdaBoost (AB), logistic regression (LR), and Gradient boosting (GB). The HCF derived from the Mordred had the dimension of 1826. Machine learning algorithms do not perform well for the high-dimensional feature vector. Therefore, both feature reduction and feature selection techniques have been used to find the optimal features. Feature reduction was accomplished by utilizing PCA, and feature selection was conducted using a sequential feature selection approach. The optimal features obtained from this process were fed to the machine learning algorithms, and the results obtained from there are provided in Table 6.7. Machine learning algorithms can also be combined to build the meta-classifier. Therefore, different meta-classifiers have also been considered in the current study, which operates in two phases. In the first phase, the optimal features (as previously stated)

were supplied to the eight machine learning algorithms that generate probability scores. In the second phase, the probabilities obtained from these machine learning classifiers were combined to create a new eight-dimensional feature vector. This feature vector was then given to machine learning algorithms in order to generate predictions. The result given by machine learning algorithms in the second phase is shown in Table 6.8. This result is superior to the one obtained without the use of a meta-classifier (Table 6.7). Figure 6.9 shows the comparison of results obtained from meta-classifiers and our proposed classifier. As can be seen from this Figure, our proposed classifier is ≈ 4-5.5, 4-5.5, 4-5.5, 7.5-10.5 % better than any of the meta classifier in terms of A_{cc}, F_s, B_a and M_{cc} values.

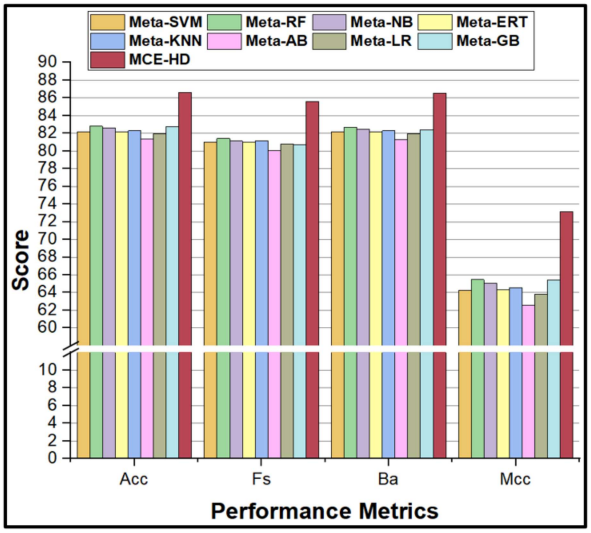

Figure 6.9: Comparison of results obtained by meta machine learning models and our proposed framework (MCE-HD).

6.4 Web server

We have five trained models (Model 1,..., Model 5) from the proposed framework. We chose the most stable model (Model with the smallest difference in test and validation performance) out of them to be the final model. This stable model (Model 4) is termed as EnDL-HemoLyt and is made available to the research community as a web server. To

serve the scientific community, we have made EnDL-HemoLyt accessible online in the form of a web server at https://endl-hemolyt.anvil.app/. The web server can provide predictions for N-terminal acetylated (implied by the presence of # at N-terminal of peptide) & C-terminal amidated (implied by the presence of + at C-terminal of peptide) peptides and can perform three activities (i) Determine if the query peptide is Hemo or Non-Hemo (ii) Do mutation analysis (iii) Run a residue scan . The details about each of the activity is given below:

- **Activity Prediction:** As shown in Figure 6.10 web server takes query peptides in raw format and returns the probability $\in$ [0,1] corresponding to each query peptide. If the probability $\in$ [0,0.5], the peptide is likely to be Non-Hemo or else if probability $\in$ (0.5,1] then the peptide is likely to be Hemo.

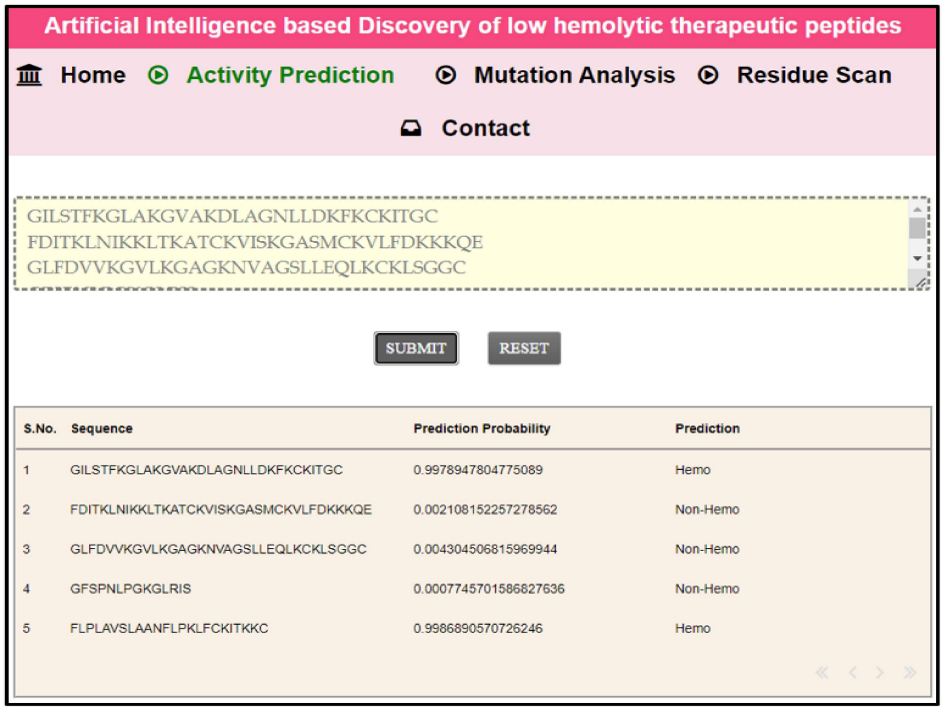

Figure 6.10: Activity prediction for query peptides.

- **Mutation Analysis:** As shown in Figure 6.11 the web server takes position and query peptide as input .The web server will then output the hemolytic activity of the query peptide and its mutants produced by replacing amino acid at the

entered location with the remaining nineteen natural amino acids.

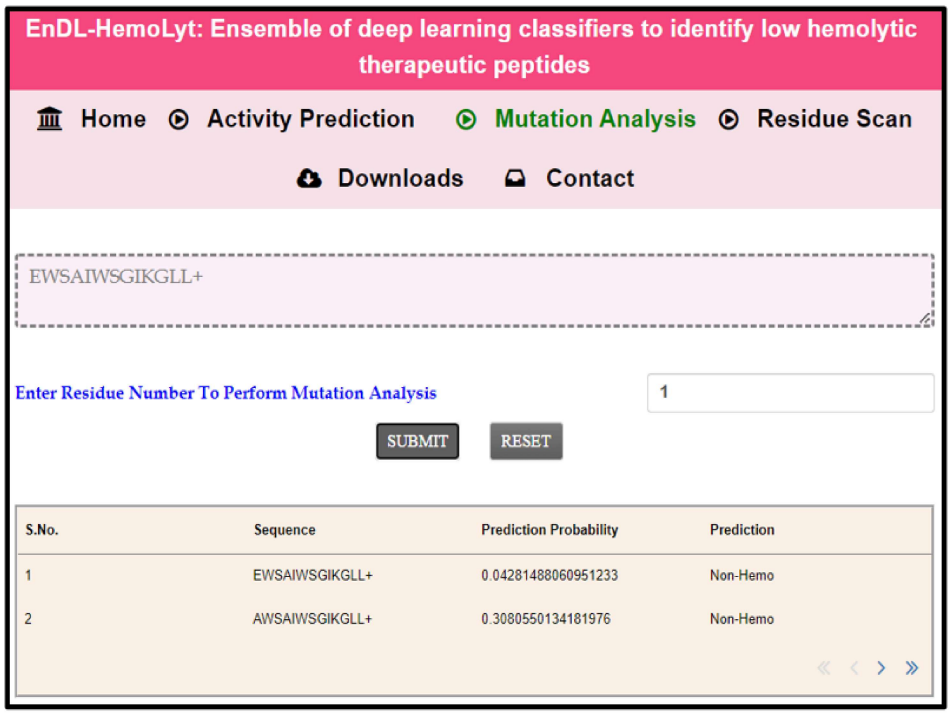

Figure 6.11: Mutation Analysis by replacing amino acid present at particular location with remaining nineteen natural amino acids.

- **Residue Scan:** As shown in Figure 6.12 web server takes amino acid and query peptide as input. The web server will then output the hemolytic activity of the query peptide and its variants constructed by substituting each amino acid (one at a time) present in the query peptide with the entered amino acid.

6.5 Conclusion

Even though therapeutic peptides have received a lot of interest as an alternative to conventional therapies, there are just a few peptide-based medications available on the market. The major roadblock to developing therapeutic peptides as medications is hemolysis, which is caused by high hemolytic peptides. As a result, it is critical to identify low hemolytic peptides while developing new peptide-based therapeutics. Experimental methods for identifying low hemolytic peptides are time-consuming, expensive,

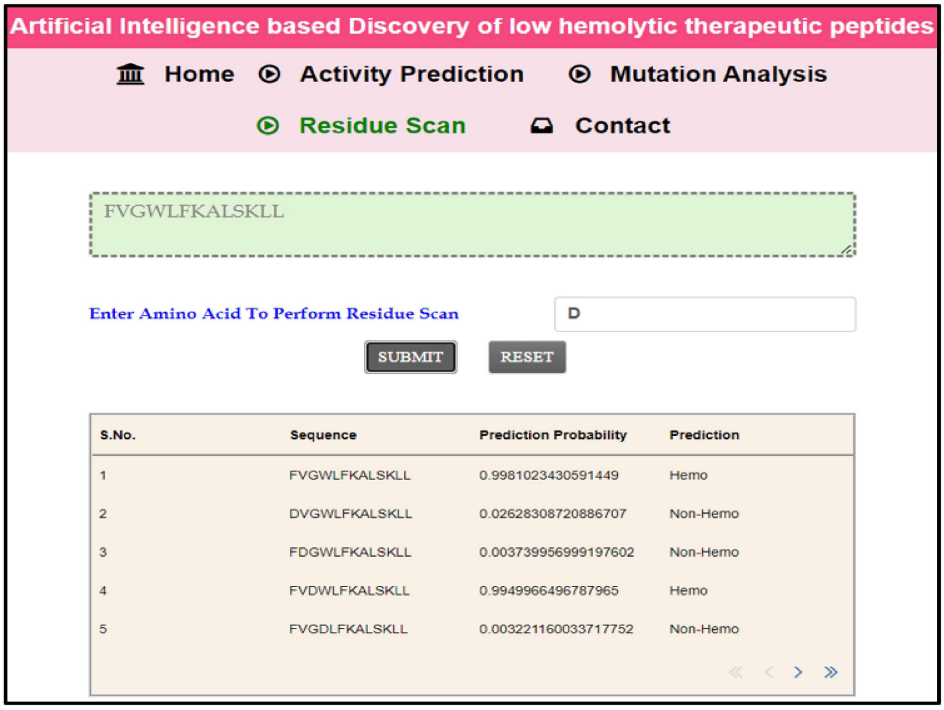

Figure 6.12: Residue Scan by substituting each amino acid (one at a time) present in the query peptide with the entered amino acid.

labor-intensive and also need to be tested on mammalian RBCs. Therefore *in-silico* tools are utilized for the initial screening of peptides to identify the low hemolytic peptides. HLPpred-Fuse, HemoPred, and HemoPI are the existing tools that can classify a given peptide as high hemolytic/low hemolytic. However, these tools have certain limitations, like (i) They cannot handle the N/C terminal modified peptides. (ii) They are not trained on hemolytic data developed recently. (iii) The performance of these tools is also low. These limitations constrict their utility. Therefore, in the current work, model named EnDL-HemoLyt has been proposed. The proposed framework can (i) Handle N/C terminal modified (N-terminal acetylated, C-terminal amidated) peptides. (ii) Trained on hemolytic data developed recently (iii) Attained the A_{cc}, S_n, P_r, F_s, S_p, B_a and Mcc values $\approx$ 87, 85, 86, 86 88, 87, and 73, respectively for the test data. Ablation studies have also been conducted in the current study, and the following conclusions have been made : (i) Aggregate prediction is always less noisy than individual prediction. Therefore, the ensemble-based model performed better than the

individual models. (ii) It is not always possible for deep learning algorithms to learn all the relevant features to perform the classification. Therefore, HCF is also required in addition to DLF. (iii) Having HCF that encodes all properties of the peptides is difficult. Therefore, DLF are also required in addition to HCF. To serve the scientific community, the model developed from the proposed framework has been made accessible online in the form of a web server at https://endl-hemolyt.anvil.app/. The web server can (i) Classify low and high hemolytic peptides better than existing tools, (ii) Provide predictions for N/C terminal modified (N-terminal acetylated, C-terminal amidated) peptides, (iii) Do mutation analysis, and (iv) Run a residue scan.

Peptide-related data is rapidly rising over time. Because of this, many more peptides will be available than there are now in a few years, which can be utilized to improve performance. However, the proposed framework is not designed to adapt to new data points. Therefore, the tool developed as a part of it will become obsolete over time. Hence, in the future, the proposed framework can be improved by considering the concept of continual learning. This will make the framework adaptive, and the tool will not become obsolete over time. N-terminal acetylation and C-terminal amidation have been considered in this study. Other work may be done in the future in which modifications other than N-terminal acetylation and C-terminal amidation can be considered.