Chapter 5

Using genetic algorithms and explainable AI for classifying and optimizing neurological peptides

The advent of the fourth industrial revolution, characterized by artificial intelligence (AI) as its fundamental component, has resulted in the mechanization of numerous manual tasks. The utilization of computational techniques has become rampant in the design and development of biopharmaceuticals. Upon conducting an analysis of the genomes of many organisms, it has been discovered that their tissues produce certain peptides that confer protection against particular diseases. The objective of this study is to design a technique that helps find neuropeptides (NPs) that can cure neurological diseases, e.g., Alzheimer's disease or Parkinson's disease. Until now, the research fraternity has merely focused on constructing classifiers for therapeutic peptides, neglecting the optimization of certain important characteristics that can help wet lab researchers optimize them for synthesis and experimental validation. Therefore, the proposed study has formulated the task of designing optimal NPs as a multi-objective optimization (MOO) problem, employing the non-dominated sorting-based genetic al-

gorithms (NSGA)-II. The model under consideration, NPpred, comprises two distinct components: NSGA-NeuroPred and BERT-NeuroPred. The former uses the NSGA-II algorithm to conduct search and modification on a population of NPs, while the latter is an explainable deep learning-based model designed to classify NPs. This model's predicted probability value and the explainable module are used in NSGA-NeuroPred. The explainable component has also led to the proposal of two novel operators: p-crossover and p-mutation. A free online application has been made accessible at the `https://neuropred.anvil.app`. Its purpose is to assist researchers in finding, optimizing, and shortlisting a set of NPs from proteins.

5.1 Introduction

The increased interest in the use and application of artificial intelligence (AI) has initiated the fourth industrial revolution, leading to the widespread advancement of machine learning (ML) and deep learning (DL). Consequently, the utilization of ML and DL techniques has been experiencing significant and rapid expansion. Protein analysis is a notable application that is being extensively studied in this regard [101, 102]. A promising area of research in this field involves the creation of an online computational tool that aims to distinguish between therapeutic and non-therapeutic peptides. Peptides, which consist of short chains of amino acid (AA) residues, play a crucial role in protein formation [44, 76]. They have the potential to facilitate the development of novel and enhanced pharmaceuticals. However, the mere construction of a model to classify and identify peptides inside proteins is inadequate for assisting wet-lab researchers in determining which peptides should be prioritized for synthesis. Additionally, excessive reliance solely on the predicted probability value given by the model for making such a decision may waste time, effort, and resources. Hence, it is imperative to develop a robust model that modifies the structure of peptides by optimizing their important characteristics, which render them favorable for chemical synthesis. To this end, a

multi-objective optimization (MOO) problem has been formulated, to be solved with the help of multi-objective optimization techniques.

The task of identifying an optimal solution in the case of MOO problems, which involve one or more competing objectives, becomes infeasible within polynomial time when the search space for potential solutions is too extensive to be explored exhaustively [103]. The aforementioned challenge prompted the creation of MOO algorithms capable of efficiently navigating the search domain within a practical timeframe. These algorithms yield satisfactory solutions that strike a reasonable balance among the specified objectives, sometimes called Pareto-optimal solutions. Some population-based meta-heuristic techniques are utilized in this regard, with the non-dominated sorting genetic algorithm (NSGA) being extensively applied among them. It generates a collection of non-dominated solutions that are Pareto-optimal with respect to a specified set of objectives. The NSGA-II [104] was introduced as an improvement over the NSGA, with its reduced time complexity and elitist nature. Therefore, this approach can be used to construct a robust framework using a specified set of objectives or properties to identify potentially therapeutic peptides within a large search domain.

Neuropeptides (NPs) are currently the subject of extensive investigation due to their potential therapeutic applications in the production of biopharmaceuticals [105] for treating neurological disorders. Recent research has demonstrated that several neuropeptides, including ghrelin, neuropeptide Y, and pituitary adenylate cyclase-activating polypeptide, exhibit neuroprotective properties in the context of neurodegenerative conditions such as Parkinson's disease and Alzheimer's disease [106]. Nevertheless, the extent of studies conducted on the automation of the identification and optimization of NPs has been restricted. In the study by [17], a meta-model called Neuropred-FRL was developed by incorporating multiple baseline models to enhance prediction performance. A stacked ensemble approach was introduced in a separate study by [107] with eight distinct machine learning algorithms at the base level. The

construction of the NeuroPred-PLM model [18] involved utilizing a multi-scale convolutional neural network. In the study conducted by [16], NPs were classified using the support vector machine (SVM) algorithm. While these models demonstrate high performance on their datasets, they rely on manually designed features as their input. This bottleneck can be addressed by employing deep learning methodologies such as recurrent neural networks (RNNs), long short-term memory networks (LSTMs) [108], temporal convolutional networks (TCNs) [52], transformers [29], bidirectional encoder representations from transformers (BERT) [109], and other similar techniques.

Moreover, the aforementioned works have concentrated on creating basic classifiers for differentiating NPs and non-NPs. When a classifier predicts a large set of NPs in a protein, the selection for further consideration is mostly based on the model's predicted probability (MPP) values. Hence, the inclusion of a human expert becomes crucial in the determination of the appropriate collection of NPs for chemical synthesis. The technique is time-intensive because wet lab researchers need to evaluate many NPs. The decision regarding peptide selection is also made based on additional properties, such as molecular weight, sequence similarity to experimentally validated peptides, and the presence of motifs (a motif is a brief pattern of three or four amino acids recurrent in many peptides of the given class). Some related work has been done in this regard. Liu et al. [110] conducted a study on applying evolutionary algorithms to identify a collection of antimicrobial peptides (AMPs). The objectives of their research encompassed the optimization of the classifier's probability value and the enhancement of search space variety. However, these two aims alone do not provide a comprehensive list of synthesizable peptides. In a study [111], the authors proposed a methodology that employs many characteristics, such as molecular weight and sequence similarity score, to assess the bioactivity of a peptide through the utilization of NSGA-II. Nevertheless, this study determines the desirability of a peptide solely based on a limited number of manually selected peptide features. Additionally, enhancing the efficacy of NPs by

modifying certain AAs in their sequences is possible, which has not been studied.

With the aim of addressing the aforementioned challenges, this chapter introduces a novel framework called NPpred which comprises two phases. During the initial phase, an explainable deep learning tool known as BERT-NeuroPred has been created [109]. This tool was specifically designed to classify NPs using BERT. Furthermore, the interpretability of the model has been achieved using the Captum library [112] to identify the amino acids (AAs) that favorably influence the classification of a given NP. The model has been trained, tuned, and tested using a dataset of experimentally validated NPs and non-NPs sourced from public databases and evaluated against many recent classifiers, namely NeuroPIpred [16], NeuroPred-FRL [17], and NeuroPred-PLM [18]. Subsequently, the MPP value provided by BERT-NeuroPred, in conjunction with properties such as molecular weight and sequence similarity score (with the NPs in the dataset) determined by the Needleman Wunsch Algorithm (NWA), has been employed to frame a new multi-objective optimization (MOO) problem. Following this, phase 2 has been executed, which finds non-dominated fronts and computes the crowding distance through the application of a preference-based function based on these objectives. Additionally, in order to maintain the motif-based information of peptides, the traditional mutation and crossover operations of NSGA-II have been adapted to introduce two novel operators known as p-mutation and p-crossover. The explainable BERT-NeuroPred model has been utilized in the case of p-mutation. The optimization model developed in the second phase of this study is referred to as NSGA-NeuroPred. When combined with BERT-NeuroPred, it forms the NPpred framework. The main contributions of this work are summarised as follows.

- A novel prediction and optimization framework (NPpred) has been introduced to identify and optimize the NPs to expedite the synthesis of biopharmaceuticals targeting neurological disorders. It consists of two distinct components, BERT-NeuroPred and NSGA-NeuroPred.

- A novel MOO problem has been framed that aims to identify a set of non-dominated NPs using the NSGA-II algorithm. The resulting framework has been named NSGA-NeuroPred.
- A novel explainable deep learning model, known as BERT-NeuroPred, has been developed to identify NPs. Comprehensive statistical performance analysis was conducted using analysis of variance (ANOVA) and the Tukey honestly significant difference (HSD) tests.
- A proposed approach involves utilizing a preference-based function to compute the crowding distance by allocating varying levels of priority to different objectives.
- To enhance the performance of NSGA-II, two new peptide-specific operators, namely p-crossover and p-mutation, have been introduced. The p-mutation operator also uses the explainable BERT-NeuroPred.
- An online application developed for the proposed model has been made available at `https://neuropred.anvil.app`. This application aims to assist researchers in the wet lab in the identification of a suitable collection of NPs based on optimal MPP value, molecular weight, and NWA score.

The rest of this chapter is organized as follows. Section 5.2 presents the tools, techniques, and dataset used in this work. The proposed work is elucidated in section 5.3. The results and outcomes have been elaborated in section 5.4. Lastly, the concluding remarks and future works are discussed in section 5.5.

5.2 Data and preliminaries

5.2.1 Dataset

The neuropeptides having 5 to 100 amino acid residues were collected from online repositories and web servers such as Neuropedia [113], DiNer [114], NeuroPep [115], NeuroPIpred [16], and Uniprot [35]. The non-NPs were collected after querying the

SwissProt database [116]. Then, the sequences in NP and non-NP sets having a sequence similarity of 90% within and across the set were removed. For this, the CD-HIT framework [94, 95, 96] was used. The final dataset (6807 NPs and 8647 non-NPs) was divided into training (80%), validation (10%), and test (10%) sets for building, tuning, and evaluating the BERT-NeuroPred model.

5.2.2 Non-dominated sorting genetic algorithm-II

NSGA is a MOO technique that aims to discover a set of Pareto-optimal solutions to a given problem within a reasonable timeframe. A MOO problem is described as per Eq. 5.1, where S denotes a set of possible solutions, f_i stands for the i^{th} objective amongst k objectives, and c_j is the j^{th} of h constraints.

$$\begin{aligned} &\textbf{Minimize :}\ f(s) = f_1(s), f_2(s), \ldots, f_k(s)|\ s\epsilon S \\ &\textbf{Subject to :}\ c_1, c_2, \ldots, c_h \end{aligned} \tag{5.1}$$

A solution $s_1\epsilon S$ is said to dominate another solution $s_2\epsilon S$ if s_1 is either better than or equivalent to s_2 in the case of every objective and is better than s_2 for at least one objective (Eq. 5.2). Moreover, a solution s_1 is labeled as Pareto-optimal if no other solution dominates it. The Pareto-optimal solutions together form a Pareto-optimal set.

$$\begin{aligned} s_2 \preceq s_1 \implies &\forall i : f_i(s_1) \leq f_i(s_2) \\ &\wedge \exists i : f_i(s_1) < f_i(s_2)\ |1 \leq i \leq k \end{aligned} \tag{5.2}$$

NSGA-II [104] is an enhancement over NSGA since it is elitist (which implies that, unlike NSGA, it keeps the best solutions obtained in every iteration) and has O (n^2) time complexity (for simple NSGA, it is O (n^3)). The implementation of NSGA-II has a series of distinct stages. Initially, a parent population of N solutions is generated. Furthermore, the process of non-dominated sorting (NDS) is executed to allocate ranks to all solutions within the population. The highest rank, rank one, is allocated to the best

non-dominated solutions, whereas rank two is designated for those that are dominated by rank one solutions. This ranking procedure continues similarly for subsequent ranks. Next, a child population is produced through the execution of crossover and mutation procedures until the population size reaches $2N$ solutions. Finally, the populations of parents and children are subjected to the selection operator, which involves comparing their ranks and crowding distance (CD) to determine if they exist in sparsely or densely populated regions inside the search space. Calculating an individual's crowding distance involves summing the distances between its adjacent solutions regarding each objective. A solution characterized by a large crowding distance indicates its placement within a region with relatively low population density. In cases where two solutions possess equal ranks, the preference is given to the solution with a higher CD. In this manner, a novel population of N solutions is chosen, and this procedure is iteratively performed for a predetermined number of generations.

5.2.3 BERT

BERT utilizes the encoder module of transformers, which facilitates the acquisition of bidirectional representations by including information from both left and right contexts [117]. It generates three types of embeddings: token embeddings, position embeddings, and segment embeddings. The token embeddings transform individual words into distinct vectors, while the position embeddings encode details regarding the relative positions of words inside a specific sequence. The segment embeddings are also utilized to maintain the association between a token and its corresponding sequence. The embeddings are entered into the encoder module, which comprises a multi-head attention (MHA) mechanism and a fully connected (FC) layer. An MHA module comprises several parallel self-attention mechanisms whose results are combined at the end before feeding to the FC layer.

5.2.4 Needleman Wunsch Algorithm

The NWA was specifically designed for utilization in bioinformatics, with the primary objective of matching biological sequences and identifying commonalities in amino acid residues [118]. This was one of the first implementations to utilize dynamic programming for comparing biological sequences. It is a widely used method for optimal sequence matching. Its purpose is to detect regions of similarity between two sequences, which can be attributed to shared structural, functional, or evolutionary relationships. The program exhaustively examines all potential alignments between a given pair of sequences and outputs the highest similarity score, representing the optimal matching between them.

5.2.5 Captum

Captum is a Python framework that offers a suite of tools designed to analyze ML/ DL models. The Captum framework facilitates the interpretation of complicated models, hence aiding in comprehending the significant or critical aspects that underlie their decision-making processes. One of the strategies offered by Captum is Integrated Gradients, which establishes a relationship between the output and the input features by integrating the gradients along a trajectory from a reference to the actual input. Regarding sequence classification, this approach assigns a positive or negative score to each component or token of the sequence based on their respective contributions to the ultimate classification outcome.

5.3 Proposed work

The model NPpred entails the construction of a deep learning classifier known as BERT-NeuroPred, which aims to discriminate between NPs and non-NPs. Then, utilizing the MPP value given by BERT-NeuroPred and two other objectives, an MOO problem

has been framed that outputs a Pareto-optimal set of NPs on resolution. The model under consideration is referred to as NSGA-NeuroPred. Both these models have been described in this section.

5.3.1 Phase 1: The BERT-NeuroPred model

A novel classifier is developed and trained to differentiate between NPs and non-NPs by taking peptide sequences as input during this stage. Several state-of-the-art ML and DL techniques were tried in isolation and in combination with one another to build an optimal classifier. A range of deep and machine learning architectures, including bi-LSTM, TCN, random forests (RF), SVM, gradient boosting (GB), and BERT, were employed to categorize NPs. The resulting models are presented as follows.

5.3.1.1 Model_BERT

The construction of a model based on BERT involved the utilization of the pre-trained BERT-large-uncased model. The system comprises 24 encoder modules that are arranged sequentially. Each of these modules is equipped with 16 attention heads. It produces an embedding with a dimensionality of 1024, which is then passed through a dense layer of 32 neurons, followed by another dense layer comprising eight neurons. A sigmoid unit in the output layer determines the final class of the peptide. Following the process of fine-tuning, the model was explained utilizing the linear integrated gradient technique of the Captum framework. A clear depiction of this model is illustrated in Figure 5.1.

5.3.1.2 Model_biLSTM

To construct the model, the initial step involved using the skip-gram approach to generate feature vectors of size +256 for every AA residue. Subsequently, these were utilized to represent each peptide, which was then fed into a bi-LSTM layer consisting of 200

units (neurons) with a drop-out rate of 10%. The feature map produced by this layer was subsequently passed into a global average pooling layer, transforming the feature map into a one-dimensional tensor. Subsequently, a pair of consecutive dense layers with dimensions 64 and 16 were used, followed by an output layer.

5.3.1.3 Model_TCN

The TCN-based model employs an identical methodology to create word embeddings as the Model_biLSTM. Every peptide, denoted by the aforementioned word embeddings, is input into a TCN residual block of 100 units (neurons). This block comprises dilations of sizes 1, 2, and 4 and is augmented by a normalization layer between the 1D convolutional layers that constitute the TCN block. The feature maps produced by this layer are subsequently input into a global average pooling layer, transforming them into a one-dimensional tensor. Subsequently, two densely connected layers with 16 and 8 neurons, respectively, followed by an output layer, were added to the architecture.

5.3.1.4 Stacked models

As elucidated in section 5.4, it was observed that Model_BERT exhibited superior performance than Model_TCN and Model_biLSTM. This model architecture (Model_BERT) was utilized as a foundation upon which further layers of deep and machine learning were incorporated to evaluate the impact on performance. The features produced by the Model_BERT framework served as input to a TCN block, Support Vector Machine (SVM), Random Forest (RF), and Gradient Boosting (GB) algorithms individually. The resulting models were denoted as Model_BERT+TCN, Model_BERT+RF, Model_BERT+SVM, and Model_BERT+GB.

5.3.2 Phase-2: The NSGA-NeuroPred framework

The NSGA-II algorithm has been employed to identify the pareto-optimal solutions, and the initial population comprises the NPs identified by BERT-NeuroPred within a

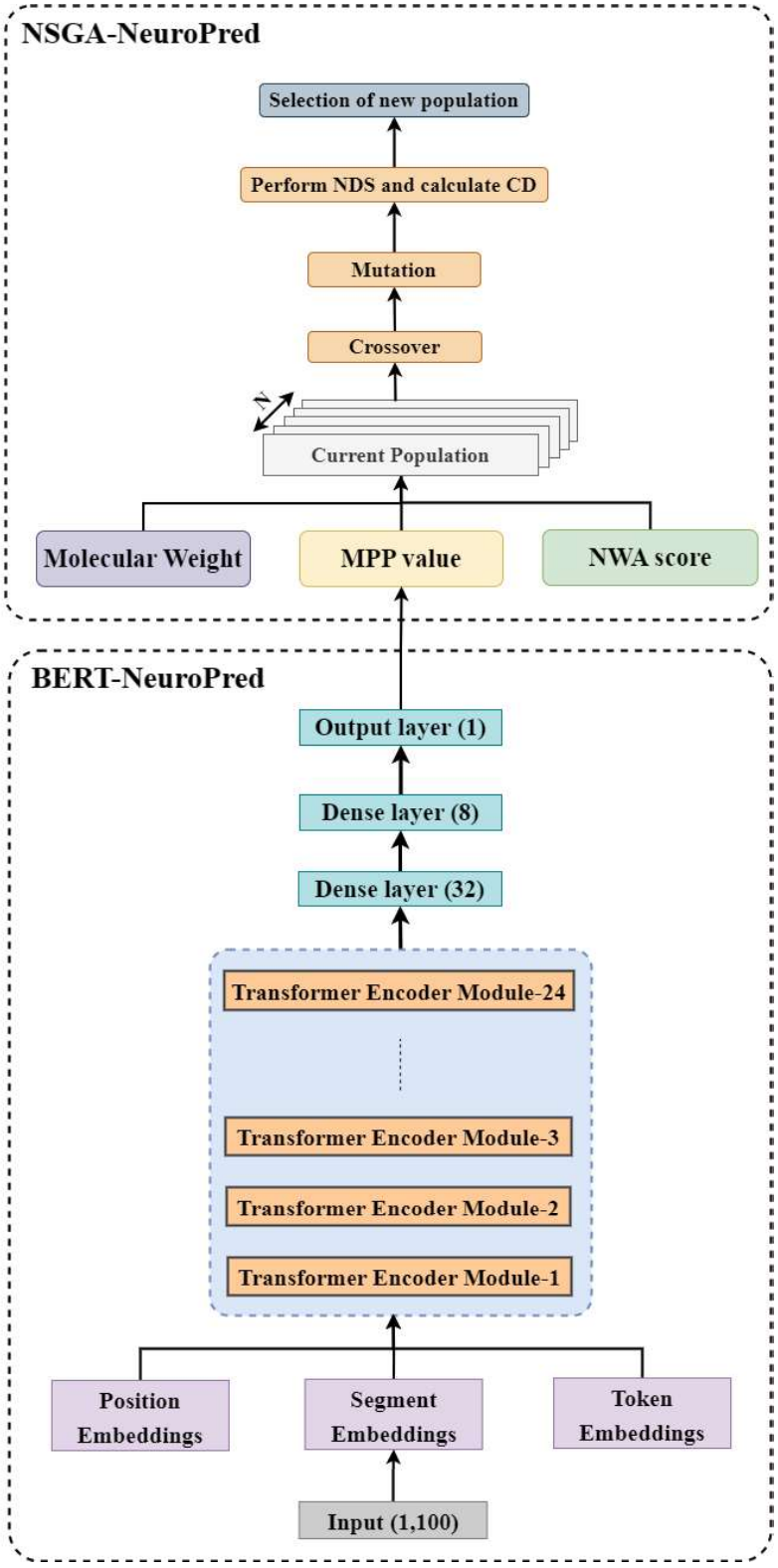

Figure 5.1: Architecture of NPpred framework

certain protein. The solutions have been encoded as peptides (sequences of letters that symbolize amino acid residues). The conducted operations and the formulated problem have been discussed as follows.

5.3.2.1 Problem formulation

This study aims to maximize three objectives: the BERT-NeuroPred model's predicted probability value (MPP), the normalized NWA score with NPs in the dataset (nw_{i_N}), and the normalized molecular weight mw_{i_N}. The objectives have been elucidated in the following order of significance.

1. **Objective-1 (Probability score given by BERT-NeuroPred Model):** The BERT-NeuroPred model gives a probability score (MPP_i) for a given peptide s_i, which signifies its estimate or likelihood of a specific peptide being neurological. Therefore, the primary aim is to maximize the MPP_i value.
2. **Objective-2 (Needleman Wunsch Algorithm score):** The NWA utilizes dynamic programming to calculate a maximum matching score between two sequences (Q and R), with lengths m and n, respectively. In this context, $Sim(Q_c, R_d)$ represents the similarity score between the sequences $Q\{1, \cdots, c\}$ and $R\{1, \cdots, d\}$, while F_{cd} denotes the NWA score for the same sequences $Q\{1, \cdots, c\}$ and $R\{1, \cdots, d\}$. The NWA score ($nw\{Q, R\}$) of a sequence Q is the maximum score achieved after comparing it with all the NPs (R) in the dataset and defined as per Eq. 5.3.

$$nw\{Q, R\} = F_{mn} \tag{5.3}$$

The NWA score (nw_i) for the i^{th} sequence of the population (s_i) is determined by Eq. 5.4. After that, the NWA score is computed with each NP in the dataset D, and the maximum of all the scores is considered the NWA score for that peptide. The min-max standardization procedure, as described in Eq. 5.5, has been employed to normalize the scores of all sequences within the population.

The minimum and maximum possible matching scores are denoted as nw_{min} and nw_{max}, respectively.

$$nw_i = \max_{j \epsilon D} nw\{s_i, s_j\} \tag{5.4}$$

$$nw_{i_N} = (nw_i - nw_{min})/(nw_{max} - nw_{min}) \tag{5.5}$$

3. **Objective-3 (Molecular weight):** According to [119], peptides characterized by a low molecular weight exhibit favorable attributes such as cost-effectiveness in synthesis and enhanced stability. Therefore, it can be inferred that peptides with low molecular weights have enhanced therapeutic success and are more feasible to produce. The formation of a peptide bond results in the liberation of a water molecule. Therefore, the molecular weight of a peptide may be calculated by adding the weights of the individual amino acid residues and subtracting the number of peptide bonds (which is one fewer than the number of amino acid residues), as shown in Equation Eq. 5.6. The standardization of molecular weight is achieved by utilizing Eq. 5.7. In this equation, the variables mw_{min} and mw_{max} represent the minimum and maximum possible weights that correspond to a peptide of a given length l.

$$mw_i = \sum_{1}^{j=l} mw_{AA_j} - 18.015 * (l - 1) \tag{5.6}$$

$$mw_{i_N} = (mw_i - mw_{min})/(mw_{max} - mw_{min}) \tag{5.7}$$

NPs possessing a high MPP value, a high NWA score, and a low molecular weight exhibit increased desirability for both production and therapeutic applications. Therefore, the NSGA-II algorithm has been employed to identify a Pareto-optimal set of NPs

based on the specified objectives. The formulation of the MOO problem is as follows.

$$\textbf{Minimize :}\ f(i) = \{(1/MPP_i), (1/nw_{i_N}), mw_{i_N}\},\ i \epsilon S \tag{5.8}$$

5.3.2.2 Selection

The selection procedure employed in NSGA-II encompasses two distinct stages: non-dominated sorting (NDS) and crowding distance computation. The process of non-dominated sorting involves partitioning the population into several fronts, each characterized by the non-dominated status of its solutions. The dominance of solutions in the first front surpasses that of individuals in the second front, and so forth. The use of crowding distance guarantees that the solutions chosen for subsequent generations exhibit diversity and are distributed evenly across the search space. The algorithm computes the separation among individuals within the population and identifies individuals situated in regions of low density to preserve diversity. Following non-dominated sorting and crowding distance calculation, the solutions are grouped together to constitute the population for the next iteration. The aforementioned procedure guarantees that NSGA-II attains convergence towards a collection of solutions exhibiting both potential and diversity.

A function based on the preference given to the aforementioned objectives has been developed to compute the crowding distance, as described by Eq. 5.9. In this approach, instead of uniformly assigning priority to all objectives, individual weights are assigned to them to calculate the crowding distance for a particular solution. In particular, there is a prioritization of the MPP value over the NWA score, which in turn is given greater importance than the molecular weight, as shown in Eqs. 5.9 and 5.10.

$$\begin{aligned} CD(i) = \alpha * (mpp_{(i+1)} - mpp_{(i-1)}) + \beta * (nw_{(i+1_N)} \\ - nw_{(i-1_N)}) + \gamma * (mw_{(i+1_N)} - mw_{(i-1_N)}) \end{aligned} \tag{5.9}$$

$$\begin{gathered} \text{where } \alpha + \beta + \gamma = 1 \\ \text{and } \alpha > \beta > \gamma > 0 \end{gathered} \tag{5.10}$$

In the given context, $mpp_{(i-1)}$ and $mpp_{(i+1)}$ denote the MPP values of the solutions immediately before and after the i^{th} solution in a sorted list of MPP values. In a sorted list of NWA values, the terms $nw_{(i-1_N)}$ and $nw_{(i+1_N)}$ denote the NWA score of the solutions immediately preceding and succeeding the i^{th} solution, respectively. Finally, $mw_{(i-1_N)}$ and $mw_{(i+1_N)}$ denote the molecular weights of the solutions immediately preceding and succeeding the i^{th} solution in a sorted sequence of molecular weights.

Algorithm 2 p-crossover and p-mutation

Input: $P_1, P_2, motifs$ ▷ *motifs* contains a list of 3 and 4 AA motifs

Output: C_1, C_2 ▷ The resultant off-springs

1: **procedure** P-CROSSOVER($P_1, P_2, motifs$)
2: $r \leftarrow$ GEN_RANDOM $(2, n-1)$ ▷ n is length of peptide
3: **while** *true* **do**
4: **if** $P_1[r, \cdots, r+2] \in motifs$ **or** $P_1[r-1, \cdots, r+1] \in motifs$ **or** $P_1[r-2, \cdots, r+1] \in motif$ **or** $P_1[r-1, \cdots, r+2] \in motifs$ **or** $P_1[r, \cdots, r+3] \in motifs$ **or** $P_2[r, \cdots, r+2] \in motifs$ **or** $P_2[r-1, \cdots, r+1] \in motifs$ **or** $P_2[r-2, \cdots, r+1] \in motifs$ **or** $P_2[r-1, \cdots, r+2] \in motifs$ **or** $P_2[r, \cdots, r+3] \in motifs$ **then**
5: $r \leftarrow$ GEN_RANDOM $(2, n-1)$
6: **else**
7: **break**
8: **end if**
9: **end while**
10: $C_1 = P_1[1, \cdots, r] \oplus P_2[r+1, \cdots, n]$

$C_2 = P_2[1, \cdots, r] \oplus P_1[r+1, \cdots, n]$

$C_1 = \text{MUTATION}(C_1)$

$C_2 = \text{MUTATION}(C_2)$

return C_1, C_2

end procedure

procedure P-MUTATION(C)

$r \leftarrow$ GEN_RANDOM $(1, n)$

$p \leftarrow$ GEN_NORMAL_RANDOM$(0, 1)$

if $p \geq 0.5$ **then**

while *true* **do**

if

$Captum\_score(C[r] > 0)$ **or** $C[r-2, \cdots, r] \; \epsilon \; motifs$ **or** $C[r-1, \cdots, r+1] \; \epsilon \; motifs$ **or** $C[r, \cdots, r+2] \; \epsilon \; motifs$ **or** $C[r-2, r+1] \; \epsilon \; motifs$ **or** $C[r-2, \cdots, r+1] \; \epsilon \; motifs$ **or** $C[r-1, \cdots, r+2] \; \epsilon \; motifs$ **or** $C[r, \cdots, r+3] \; \epsilon \; motifs$ **then**

$r \leftarrow$ GEN_RANDOM $(1, n)$

else

break

end if

end while

$list \leftarrow$ **empty**

for $i = 1$ **to** 20 **do**

$aa \leftarrow AA[i]$

if

$C[r-2, \cdots, r-1] \oplus aa \; \epsilon \; motif$ **or** $C[r-1] \oplus aa \oplus C[r+1] \; \epsilon \; motifs$ **or** $C[r] \oplus C[r+1] \oplus aa \; \epsilon \; motifs$ **or** $C[r-3, \cdots, r-1] \oplus aa \; \epsilon \; motifs$ **or** $C[r-2, r-1] \oplus aa \oplus C[r+1] \; \epsilon \; motifs$ **or** $C[r-2, r-1] \oplus aa \oplus C[r+1] \; \epsilon \; motifs$ **or** $C[r-1] \oplus$

$aa \oplus C[r+1, r+2] \; \epsilon \; motifs$ **or** $aa \oplus C[r+1, \cdots, r+3] \; \epsilon \; motifs$ **then**

31: $list = list \oplus aa$

32: **end if**

33: **end for**

34: **if** $list =$ **empty then**

35: $x \leftarrow$ GEN_RANDOM $(1, 20)$

36: $C[r] = AA[i]$

37: **else**

38: $x \leftarrow$ GEN_RANDOM $(1, len(list))$

39: $C[r] = list[x]$

40: **end if**

41: **end if**

42: **return** C

43: **end procedure**

5.3.2.3 p-crossover

Typically, a single point crossover operation entails the random selection of a crossover point (CP) and the subsequent exchange of the substrings located to the right of the CP between the parent peptides, as indicated in lines 10-11 of Algorithm 2. In this study, a modification has been made to the crossover operator, and a new operator called p-crossover has been introduced. The p-crossover operator avoids performing crossover at a selected CP if it falls in the middle of a three or four-amino acid motif, as indicated in lines 3-7. Instead, the algorithm continues searching for an alternative CP until it identifies one that does not interfere with any existing motifs in any parent.

5.3.2.4 p-mutation

The conventional mutation operator has been adapted and redefined in this study as a novel operator referred to as p-mutation. The operation of p-mutation entails the evalu-

ation of two conditions. Consider a scenario where a certain position r is chosen for the mutation or replacement. As indicated in lines 20-23, the AA to be replaced is excluded from mutation if it has a favorable impact on the categorization by BERT-NeuroPred or if it is a component of a motif with a length of three or four. The method iteratively explores alternative mutation positions until it identifies one that neither disrupts a motif nor provides a favorable contribution to the NP classification. Subsequently, as elucidated in lines 30-32, the algorithm examines the possibility of substituting the amino acid residue located at position r with another residue to generate a three or four-AA motif inside the peptide. If the aforementioned possibility does not exist, the amino acid (AA) located at position r is substituted with an AA selected at random (line 38).

5.4 Experiments, results, and discussions

Extensive experimentation has been conducted on the proposed NPpred framework, which was coded using Python. The trials are conducted on a GPU node with an NVIDIA V100 core. The compute node also included 16 GB of random access memory. In order to execute the deep learning model, the Keras framework with Tensorflow was employed as the platform [40].

5.4.1 Evaluation Criteria

Several evaluation metrics, like accuracy, f1-score, and area under the receiver operating characteristic curve (AUC), were used to evaluate the proposed model. These metrics are as described in Eqs. 5.11-5.16.

$$Accuracy\ (Acc) = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.11}$$

$$Precision\ (Pr) = \frac{TP}{TP + FP} \tag{5.12}$$

$$\textit{Recall } (Rec) \text{ (or } \textit{True Positive Rate } (TPR)) = \frac{TN}{TN + FP} \tag{5.13}$$

$$\textit{F1-score } (Fs) = \frac{2 \times Pr \times Rec}{Pr + Rec} \tag{5.14}$$

$$\textit{False Positive Rate } (FPR) = 1 - \frac{TN}{FP + TN} \tag{5.15}$$

$$AUC = \int TPR.\ \mathrm{d}(FPR) \tag{5.16}$$

5.4.2 Analysis of phase-1

This study presents a series of experiments conducted to construct models utilizing biLSTM, TCN, and BERT architectures, with the objective of identifying the most accurate model. In addition, the features generated by the Model_BERT have been used with many machine/deep learning methods, resulting in the training and evaluation of numerous stacked models. The models were evaluated and compared using their average performance on the test set. The statistical significance of the acquired results has been assessed by implementing a comprehensive analysis of variance (ANOVA) test. Subsequently, a post-hoc analysis is conducted employing the Tukey honestly significant difference (HSD) test. The null hypothesis posits no significant difference in the mean performance of all the classifiers. If the p-value obtained from the ANOVA is smaller than the predetermined significance level (0.05), it can be concluded that the null hypothesis is rejected. This indicates a statistically significant difference in the mean performance of at least one classifier in comparison to others. The Tukey HSD test is conducted to identify the pair of classifiers with statistically significant differences in mean performance, which is determined by comparing the adjusted p-value of the pairs to the predetermined significance level (0.05). Additionally, the 95% confidence interval (CI), which is calculated based on the test results and consists of a lower bound (LB) and an upper bound (UB), is examined to ensure that it does not include zero. If the value of the mean difference (meandiff) between the first and second classifiers is

negative, it indicates that the former performs better than the latter.

5.4.2.1 Deep learning models

This study involved conducting comprehensive experiments by training and fine-tuning three deep learning algorithms, BERT, TCN, and biLSTM, using the training and validation sets. The obtained models have been evaluated by comparing their performance on the test set, as depicted in Figure 5.2. The results indicate that Model_BERT outperforms the other models on all metrics. To assess the statistical significance of the findings, the ANOVA and Tukey HSD tests are conducted on the results obtained from the models (presented in Tables 5.1-5.3). The obtained p-value for all three metrics is less than the predetermined significance level of 0.05. So, the observed disparity in the average performance across all three models exhibits statistical significance. Moreover, while conducting the Tukey HSD test, it has been observed that the adjusted p-value for all pairs of models is less than 0.05. The Model_BERT exhibits the highest mean performance across all metrics, leading to the conclusion that it is the best among the three models. This may be due to the fact that BERT is pretrained on a large corpus, making it easier for it to achieve better results than its other DL counterparts like TCN or LSTM.

5.4.2.2 Stacked models

To conduct a more comprehensive analysis, some ML/ DL-based layers have been used in front of the underlying Model_BERT architecture to assess whether any notable enhancements in performance could be achieved. For this, algorithms such as TCN, RF, GB, and SVM have been employed, and the resulting models were designated as Model_BERT+TCN, Model_BERT+RF, Model_BERT+GB, and Model_BERT+SVM. These models are compared on the test set, and their respective performances have been illustrated in Figure 5.3. The results indicate that the performance of the Model_BERT

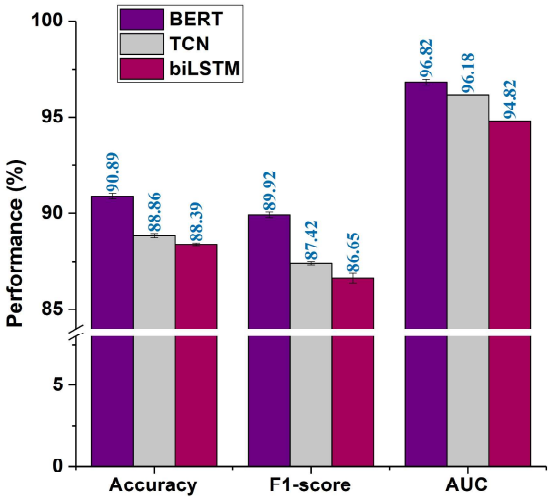

Figure 5.2: Performance of different deep learning models on the test set

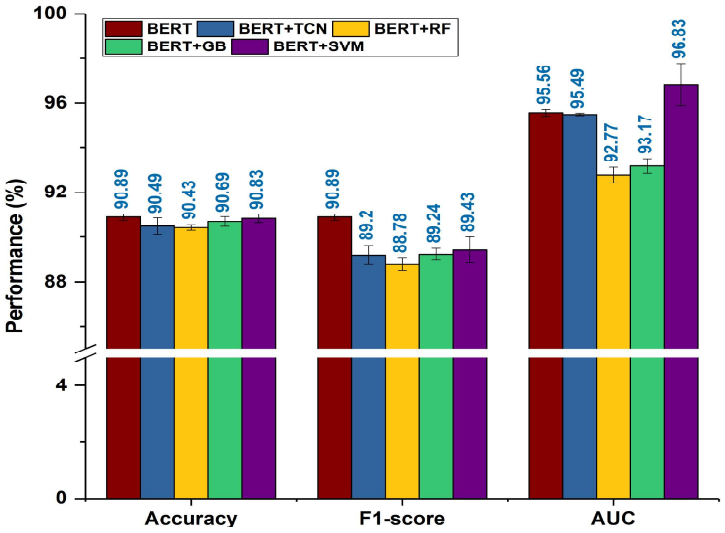

Figure 5.3: Performance of different stacked models on the test set

Table 5.1: ANOVA on accuracy (%) of all the proposed models after training on ten different dataset splits

(a) Input summary

Group	Count	Sum	Average	Variance
Model_BERT	10	908.89	90.89	0.15
Model_TCN	10	883.98	88.39	0.08
Model_biLSTM	10	888.59	88.86	0.11

(b) ANOVA result

Source of Variation	SS	df	MS	F-stat	P-value	F-crit
Between Groups	35.13	2	17.56	153.52	1.79E-15	3.35
Within Groups	3.09	27	0.11	-	-	-
Total	38.22	29	-	-	-	-

(c) Tukey HSD result

Group 1	Group 2	Meandiff	p-adjusted	Lower-bound	Upper-Bound
Model_BERT	Model_TCN	-2.03	0.00	-2.41	-1.65
Model_BERT	Model_biLSTM	-2.49	0.00	-2.87	-2.12
Model_TCN	Model_biLSTM	-0.46	0.01	-0.84	-0.09

Table 5.2: ANOVA on f1-score (%) of all the proposed models after training on ten different dataset splits

(a) Input summary

Group	Count	Sum	Average	Variance
Model_BERT	10	899.18	89.92	0.15
Model_TCN	10	874.16	87.42	0.09
Model_biLSTM	10	866.50	86.65	0.24

(b) ANOVA result

Source of Variation	SS	df	MS	F-stat	P-value	F-crit
Between Groups	58.42	2	29.21	178.56	2.71E-16	3.35
Within Groups	4.42	27	0.16	-	-	-
Total	62.84	29	-	-	-	-

(c) Tukey HSD result

Group 1	Group 2	Meandiff	p-adjusted	Lower-bound	Upper-Bound
Model_BERT	Model_TCN	-2.50	0.00	-2.95	-2.05
Model_BERT	Model_biLSTM	-3.27	0.00	-3.72	-2.82
Model_TCN	Model_biLSTM	-0.77	0.00	-1.21	-0.32

+SVM is superior to that of the Model_BERT in terms of AUC. However, no other model demonstrates superior performance compared to the Model_BERT. The experiments revealed that incorporating various deep learning layers onto the Model_BERT

Table 5.3: ANOVA on AUC (%) of all the proposed models after training on ten different dataset splits

(a) Input summary

Group	Count	Sum	Average	Variance
Model_BERT	10	968.16	96.82	0.15
Model_TCN	10	961.81	96.18	0.01
Model_biLSTM	10	948.22	94.82	0.01

(b) ANOVA result

Source of Variation	SS	df	MS	F-stat	P-value	F-crit
Between Groups	20.75	2	10.37	181.53	2.20E-16	3.35
Within Groups	1.54	27	0.06	-	-	-
Total	22.29	29	-	-	-	-

(c) Tukey HSD result

Group 1	Group 2	Meandiff	p-adjusted	Lower-bound	Upper-Bound
Model_BERT	Model_TCN	-0.64	0.00	-0.90	-0.37
Model_BERT	Model_biLSTM	-1.99	0.00	-2.26	-1.73
Model_TCN	Model_biLSTM	-1.36	0.00	-1.62	-1.09

Table 5.4: ANOVA on accuracy (%) of all the proposed models after training on ten different dataset splits

(a) Input summary

Group	Count	Sum	Average	Variance
Model_BERT	10	908.89	90.89	0.15
Model_BERT+TCN	10	904.88	90.49	0.36
Model_BERT+RF	10	904.34	90.43	0.12
Model_BERT+GB	10	906.96	90.69	0.21
Model_BERT+SVM	10	908.25	90.83	0.19

(b) ANOVA result

Source of Variation	SS	df	MS	F-stat	P-value	F-crit
Between Groups	29.34	5	5.87	7.88	1.25E-05	2.39
Within Groups	40.22	54	0.74	-	-	-
Total	69.56	59	-	-	-	-

(c) Tukey HSD result

Group 1	Group 2	Meandiff	p-adjusted	Lower-bound	Upper-Bound
Model_BERT	Model_BERT+GB	-0.19	0.87	-0.76	0.37
Model_BERT	Model_BERT+RF	-0.45	0.17	-1.02	0.11
Model_BERT	Model_BERT+SVM	-0.06	0.99	-0.63	0.50
Model_BERT	Model_BERT+TCN	-0.40	0.28	-0.97	0.17

Table 5.5: ANOVA on F1-score (%) of all the proposed models after training on ten different dataset splits

(a) Input summary

Group	Count	Sum	Average	Variance
Model_BERT	10	899.18	89.92	0.15
Model_BERT+TCN	10	892.01	89.20	0.42
Model_BERT+RF	10	887.83	88.78	0.29
Model_BERT+GB	10	892.38	89.24	0.26
Model_BERT+SVM	10	893.41	89.34	0.57

(b) ANOVA result

Source of Variation	SS	df	MS	F-stat	P-value	F-crit
Between Groups	6.64	4	1.66	4.89	0.002	2.58
Within Groups	15.25	45	0.34	-	-	-
Total	21.89	49	-	-	-	-

(c) Tukey HSD result

Group 1	Group 2	Meandiff	p-adjusted	Lower-bound	Upper-Bound
Model_BERT	Model_BERT+GB	-0.68	0.08	-1.42	0.06
Model_BERT	Model_BERT+RF	-1.138	0.00	-1.88	-0.39
Model_BERT	Model_BERT+SVM	-0.58	0.19	-1.32	0.16
Model_BERT	Model_BERT+TCN	-0.72	0.06	-1.46	0.02

resulted in minimal enhancement/ decline in its performance, as depicted in Figure 5.3. Compared to Model_BERT, Model_BERT+SVM has superior performance in terms of AUC while exhibiting inferior performance in terms of f1-score and accuracy. In the case of all performance metrics, it has been observed that other models exhibit either comparable or inferior performance compared to Model_BERT. To determine the necessity of adding extra layers to the baseline model, an examination was conducted to assess if there exists a statistically significant disparity in the average performances of these models compared to the Model_BERT. The findings are presented in Tables 5.4-5.6 and are subsequently analyzed in the following manner.

1. **Accuracy**: The adjusted p-value, as indicated in Table 5.4 (b), is less than the significance level of 0.05. This suggests a statistically significant difference in the mean performances of some models. However, the means of all the stacked

Table 5.6: ANOVA on AUC (%) of all the proposed models after training on ten different dataset splits

(a) Input summary

Group	Count	Sum	Average	Variance
Model_BERT	10	968.16	96.82	0.15
Model_BERT+TCN	10	962.33	96.23	0.05
Model_BERT+RF	10	927.66	92.77	0.34
Model_BERT+GB	10	931.69	93.17	0.33
Model_BERT+SVM	10	968.25	96.83	0.93

(b) ANOVA result

Source of Variation	SS	df	MS	F-stat	P-value	F-crit
Between Groups	163.59	4	40.89	113.35	6.81E-23	2.58
Within Groups	16.24	45	0.36	-	-	-
Total	179.83	49	-	-	-	-

(c) Tukey HSD result

Group 1	Group 2	Meandiff	p-adjusted	Lower-bound	Upper-Bound
Model_BERT	Model_BERT+GB	-3.65	0.00	-4.41	-2.88
Model_BERT	Model_BERT+RF	-4.05	0.00	-4.81	-3.29
Model_BERT	Model_BERT+SVM	0.01	1.00	-0.75	0.77
Model_BERT	Model_BERT+TCN	-0.58	0.21	-1.35	0.18

models are equivalent to that of Model_BERT, as evidenced by the adjusted p-value exceeding 0.05 in all instances.

2. **F1-score**: The p-value, in this case, is less than 0.05, as indicated in Table 5.5 (b). Moreover, the Tukey test provides evidence that the disparity in means between Model_BERT and both Model_BERT+TCN and Model_BERT+SVM is statistically significant. In comparison to Model_BERT+TCN, the performance of Model_BERT is superior. In contrast, its performance is inferior to that of the Model_BERT+SVM. Regarding the remaining models, it is observed that the performance of Model_BERT is almost on par with them.
3. **AUC**: The p-value, in this case, is lesser than 0.05, as seen in Table 5.6 (b). Moreover, the Tukey test provides evidence to support the claim that there is a statistically significant difference in the means between Model_BERT and both Model_BERT+TCN and Model_BERT+SVM. Compared to Model_BERT+TCN, the performance of Model_BERT is superior. In contrast, its performance is

Table 5.7: Comparison of BERT-NeuroPred with state-of-the-art (SOTA) models.

Model	Accuracy(%)	F1-score(%)	AUC(%)
BERT-NeuroPred	**91.14**	**90.08**	**97.12**
Neuropred-PLM	88.24	87.96	88.95
NeuroPred-FRL	87.27	84.83	93.84
NeuroPIpred	61.82	53.86	60.69

inferior to that of the Model_BERT+SVM. Regarding the remaining models, it is observed that the performance of Model_BERT is approximately similar to them. Since BERT is fully capable of generating rich feature representations on its own, addition of other ML or DL layers on top of the BERT's output will cause only a little variation in the results. Hence, as the performance of the baseline model (Model_BERT) did not exhibit a substantial enhancement following the integration of diverse machine/deep learning layers, it has been ultimately designated as BERT-NeuroPred and employed for the classification of NPs. Subsequently, the outcomes of the proposed model are elucidated using the Captum framework for utilization by the p-mutation operator. In addition, this model has been evaluated against several recent NP classifiers, including NeuroPIpred [16], NeuroPred-FRL [17], and NeuroPred-PLM [18], using the test dataset. The findings in Table 5.7 and Figure 5.4 indicate that BERT-NeuroPred outperforms the aforementioned models.

5.4.3 Analysis of phase-2

The second phase involves the deployment of a web application at `https://neuropred.anvil.app`. This application facilitated the identification of 200 neuropeptides, meeting the criteria of having a minimum MPP value of 0.90 and consisting of 20 amino acids. These neuropeptides were extracted from three widely recognized neuroproteins from the UniProt database. The peptides have been used as the initial population for NSGA-NeuroPred, which is executed for 50 iterations, yielding a Pareto-optimal set of NPs in the end. The visualization of peptides (the NPs in the original dataset, the

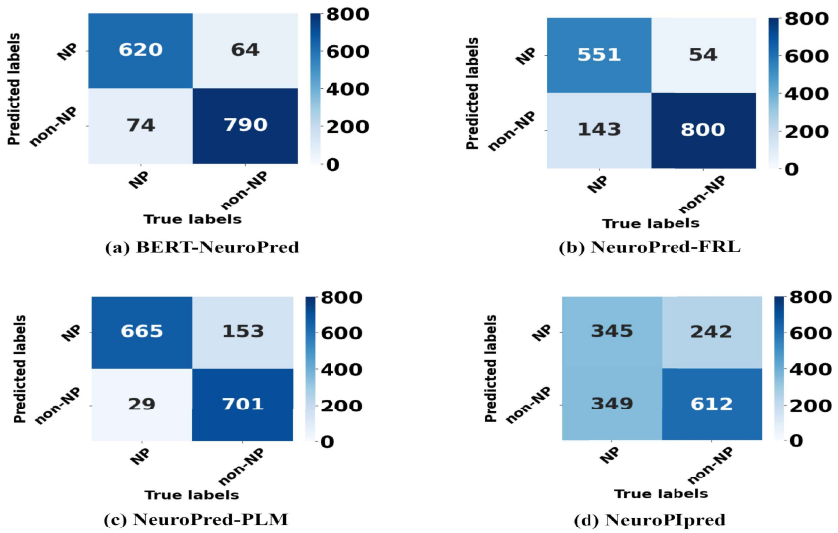

Figure 5.4: Confusion matrix depicting performance of the proposed and state-of-the-art models on the test set

NPs identified by BERT-NeuroPred with MPP values greater than or equal to 0.9, and the NPs present in the Pareto-optimal set discovered by NSGA-NeuroPred (Figure 5.5)) is achieved by applying the isometric mapping approach. The distribution of all three aforementioned groups of NPs exhibits a notable overlap. The observation of interest is that the Pareto-optimal NPs, determined by NSGA-NeuroPred, are spread out across the distribution while also being focused in the central region of the distribution of empirically validated NPs. The findings indicate that the implementation of NSGA-NeuroPred yields enhanced outcomes. Additionally, it offers a reduced and refined collection of NPs to wet lab researchers, facilitating faster chemical synthesis and experimental validation.

5.5 Conclusion

This study presents a novel framework that combines explainable deep learning and genetic algorithms to expedite the process of discovering and synthesizing neurological peptides. In addition to constructing a classifier, the proposed framework includes

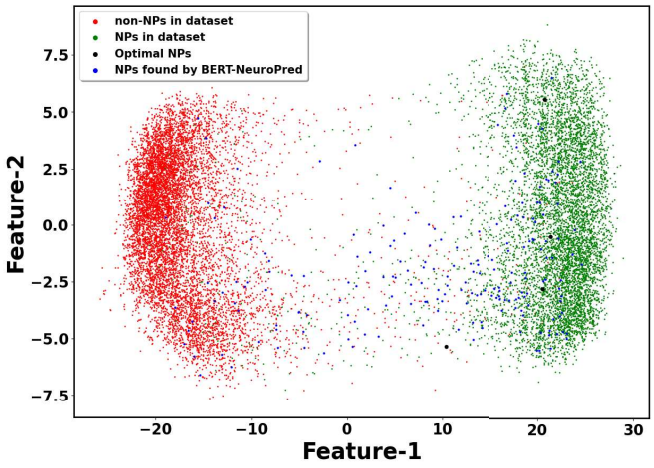

Figure 5.5: Isometric mapping of peptides predicted by NPpred

modeling a multi-objective optimization problem incorporating molecular weight and resemblance to experimentally validated NPs as objectives to give a Pareto-optimal set of NPs. This included the investigation of the search space of NPs identified by the classifier in various proteins. During the first step, a classifier known as BERT-NeuroPred is developed to distinguish between NPs and non-NPs. The MPP value provided was used with the NWA score and molecular weight to formulate a minimization problem, and the MOO module of the framework has been designated as NSGA-NeuroPred. Additionally, two unique operators, namely p-crossover and p-mutation, were introduced based on motifs and the explainable BERT-NeuroPred model. The amalgamation of the BERT-NeuroPred and NSGA-NeuroPred modules has been designated as NPpred. An online application, freely accessible at the `https://neuropred.anvil.app`, has been developed utilizing this framework. To conduct a preliminary investigation of this application, three widely recognized neuroproteins were input into it, resulting in the identification, and modification of Pareto-optimal NPs. The NPs in the Pareto-optimal front were observed to be situated inside the central region of the annotated NPs. These NPs have the potential to be chemically produced and subjected successfully for the testing of neurological activity.

In further studies, additional secondary and tertiary attributes of peptides, such as isoelectric point and amphipathicity, may be included as optimization objectives. Moreover, it is worth investigating the identification of suitable neuropeptides that have the ability to rapidly form a conjugate with drug carriers, such as blood-brain barrier penetrating peptides which would facilitate the development of bio-pharmaceuticals that can efficiently reach their intended targets within the brain. Finally, the exploration of alternative sequence modeling techniques to construct classifiers that are superior in terms of both performance and efficiency can be undertaken.