Mathematical Biosciences and Engineering

MBE, 20(12): 21563–21587.
DOI: 10.3934/mbe.2023954
Received: 25 September 2023
Revised: 15 November 2023
Accepted: 30 November 2023
Published: 05 December 2023

http://www.aimspress.com/journal/MBE

*Research article*

# iAVPs-ResBi: Identifying antiviral peptides by using deep residual network and bidirectional gated recurrent unit

**Xinyan Ma[1], Yunyun Liang[1,*] and Shengli Zhang[2,*]**

[1] School of Science, Xi'an Polytechnic University, Xi'an 710048, China
[2] School of Mathematics and Statistics, Xidian University, Xi'an 710071, China

* **Correspondence:** Email: yunyunliang88@163.com, shengli0201@163.com.

**Abstract:** Human history is also the history of the fight against viral diseases. From the eradication of viruses to coexistence, advances in biomedicine have led to a more objective understanding of viruses and a corresponding increase in the tools and methods to combat them. More recently, antiviral peptides (AVPs) have been discovered, which due to their superior advantages, have achieved great impact as antiviral drugs. Therefore, it is very necessary to develop a prediction model to accurately identify AVPs. In this paper, we develop the iAVPs-ResBi model using k-spaced amino acid pairs (KSAAP), encoding based on grouped weight (EBGW), enhanced grouped amino acid composition (EGAAC) based on the N5C5 sequence, composition, transition and distribution (CTD) based on physicochemical properties for multi-feature extraction. Then we adopt bidirectional long short-term memory (BiLSTM) to fuse features for obtaining the most differentiated information from multiple original feature sets. Finally, the deep model is built by combining improved residual network and bidirectional gated recurrent unit (BiGRU) to perform classification. The results obtained are better than those of the existing methods, and the accuracies are 95.07, 98.07, 94.29 and 97.50% on the four datasets, which show that iAVPs-ResBi can be used as an effective tool for the identification of antiviral peptides. The datasets and codes are freely available at https://github.com/yunyunliang88/iAVPs-ResBi.

**Keywords:** antiviral peptides; features extraction; feature fusion; residual network; bidirectional gated recurrent unit

## 1. Introduction

Viruses are microscopic infectious complexes that replicate in host cells. As a type of cellular organism, viruses have a very simple structure and do not have their own metabolic system, which means they have to colonize living cells [1]. As a result, once the virus has left the host cell, it loses all its vital activities and can no longer replicate, but it is highly pathogenic [2]. The great diversity of viruses, their genetic variation, their unclear routes of transmission and their tendency to acquire specific resistance to antiviral drugs through the evolution of mutations make it extremely difficult for researchers to develop effective and safe specific treatments [3,4]. Infectious diseases caused by viral pathogens have a major impact worldwide. Zoonotic viruses such as Ebola, Zika fever, West Nile virus, HIV, SARS-CoV and SARS-CoV-2 are particularly dangerous. COVID-19, in particular, has caused millions of deaths worldwide. The development of antiviral drugs is therefore essential [5].

Peptides have interesting pharmacological properties such as high selectivity and relative safety [6]. Compared to conventional non-peptide drugs, these potential antiviral agents have a number of advantages. Peptides are highly specific, readily present in the human body and perform a wide range of biological functions. Furthermore, peptides are inexpensive to produce and easy to modify and synthesize [7]. They are mainly used as signaling and regulatory molecules in various physiological processes [8]. Previously, antiviral peptides (AVPs) are isolated from animal excretions after activation of the host defense process. Today, however, they are now available by reasonable design, either chemically [9] or from recombinant libraries. [10]. AVPs can be divided into two categories according to their mode of action on the host organism: virus-targeted and peptide-targeted [11]. To inactivate specific viral proteins [12], virus-targeted peptides inhibit viral enzymes involved in transcription and replication [13,14]. Only a few compounds with antiviral activity are known. AVPs are a kind of small polypeptide molecules with biological activity, which can kill or inhibit viruses. And they have been shown experimentally to prevent viruses from attaching to and invading host cells [15,16]. The unique molecular structure and mechanism of action have made antiviral peptides a hot spot in antiviral research. The majority of antiviral drugs act on specific regions or components of the virus to inhibit its growth. New antiviral drugs are being discovered by targeting various stages of the virus's life cycle, such as the process by which the virus enters the host and the process by which it is synthesized within the host [17].

The limited availability of therapeutic molecules targeting many viral infections means that new antiviral drug candidates need to be found to control resurgent and drug-resistant pathogenic viruses [18]. Experimentally validated antiviral peptides can therefore be used as an alternative strategy against medically important viruses [19]. Machine learning algorithms, in particular deep learning algorithms, have been used to efficiently identify antiviral peptides. Thakur et al. [20] developed the first anti-viral peptide prediction tool, called AVPpred, using amino acid composition and physicochemical properties and support vector machine. Chang et al. [21] showed that a physicochemical model using random forests was better at identifying antiviral peptides. Zare et al. [22] classified antiviral peptides using pseudo amino acid composition (PseAAC) and Adaboost. Lissabet et al. [23] developed AntiVPP 1.0 by employing random forest algorithm to predict antiviral peptides based on net charge, number of hydrogen bond donors, molecular weight and hydrophilicity index. Schaduangrat et al. [24] used different machine learning algorithms and proposed Meta-iAVP, which can extract efficient feature representations based on prediction parameters obtained from feature types. Chowdhury et al. [25] proposed Firm-AVP based on the physicochemical and structural properties of amino acid sequences.

Li et al. [26] proposed DeepAVP, a two-channel deep neural network integration model for the analysis of variable lengths of antiviral peptides. Pang et al. proposed a two-stage classification scheme and established PreAntiCoV [27] and AVPIden [28] models. Timmons et al. [29] proposed ENNAVIA, a neural network-based antiviral peptide identification model. Agarwal et al. [30] elaborated on the identification and validation of antiviral peptides. Charoenkwan et al. [31] summarized machine learning methods for virus detection directly from sequence information. Manavalan et al. [32] evaluated highly specific virus predictors. Kurata et al. proposed a method called iACVP [33] and used conventional features, binary profiling and word2vec (W2V) embedding to detect coronavirus antigenic peptides (ACVP). Recent studies have collected AVPs from the literature and published them in databases such as AVPdb [3], DBAASP [34], CAMP [35] and APD3 [36]. These databases form the basis of computational predictors that help researchers identify AVPs, save time and reduce labor costs. A number of prediction tools have been developed to identify AVPs, but they still have certain limitations: they only support the calculation of peptides in a certain range, only some models use deep learning and the accuracy of the prediction needs to be improved.

In this paper, the iAVP-ResBi model is proposed to identify AVPs. Firstly, to reflect the information of AVPs more comprehensively, we use the k-spaced amino acid pairs (KSAAP), enhanced grouped amino acid composition (EGAAC) based on the N5C5 sequence, encoding based on grouped weight (EBGW), and composition, transition and distribution (CTD) to extract features. Then, the four features groups are fused by bidirectional long short term memory (BiLSTM). After that, the fused features are input into deep algorithm framework to identify AVPs. The deep learning framework is constructed by combining improved residual neural network and bidirectional gated recurrent neural network (BiGRU). Residual neural network is a kind of shortcut connection network that prevents gradients from exploding and disappearing. We adopt two residual blocks, each consisting of batch normalization (BN) layer, and Relu activation function and two convolution layers. After processing by the residual neural network, the feature information is entered into BiGRU. Unnecessary information is deleted and useful information is retained by means of update and reset gates. Then, we add the softmax activation layer for classification. 5-fold cross-validation and independent validation are used for verifying the generalizability of the model. The results indicate that, compared with previous studies, our model achieves optimal results and significantly outperforms existing models, and the iAVP-ResBi model can accurately identify AVPs. In order to better explain the model constructed, we draw a flowchart of the model building process, as shown in Figure 1.

## 2. Materials and methods

### *2.1. Datasets*

In this paper, to facilitate comparison with existing models, we select the well-designed AVP datasets created by Thakur et al. [20]. They extract 1245 peptide sequences from more than 80 relevant papers and patents containing more than 30 types of antiviral activity sequences like HIV, HCV, HSV, RSV, SARS-CoV, influenza, etc., which contain duplicate peptides or identical peptides. Nearly 91% of peptides come from nature, and the rest can be obtained by artificial synthesis. After removing the duplicate peptides from 1245 peptide sequences by sequence alignment, 604 high-efficiency AVPs and 452 least or non-effective AVPs are finally obtained, which are randomly divided into training set $T^{544p+407n}$ (544 positive and 407 negative for cross-validation) and test set $V^{60p+45n}$ (60 positive and 45 negative for the independent validation set), respectively, according to the proportion of 90

and 10%.

In an earlier antimicrobial peptide prediction model [37], a new dataset is constructed using non-experimental negative peptides instead of experimentally validated negative peptides in order to design a better model and better test the generalizability of the model. Therefore, Thakur et al. [20] construct the other two datasets based on non-experimental negative peptides: the training set $T^{544p+544n}$ (544 positive and 544 negative) and independent validation set $V^{60p+60n}$ (60 positive and 60 negative).

In order to further test the generalizability and ensure the reliability of the model, dataset ENNAVIA-C constructed by Timmons and Hewage [29] is adopted for identification of anti-coronavirus peptides. ENNAVIA-C includes 109 peptide sequences with anti-coronavirus activity and 356 peptide sequences with experimentally validated poor or no antiviral activity.

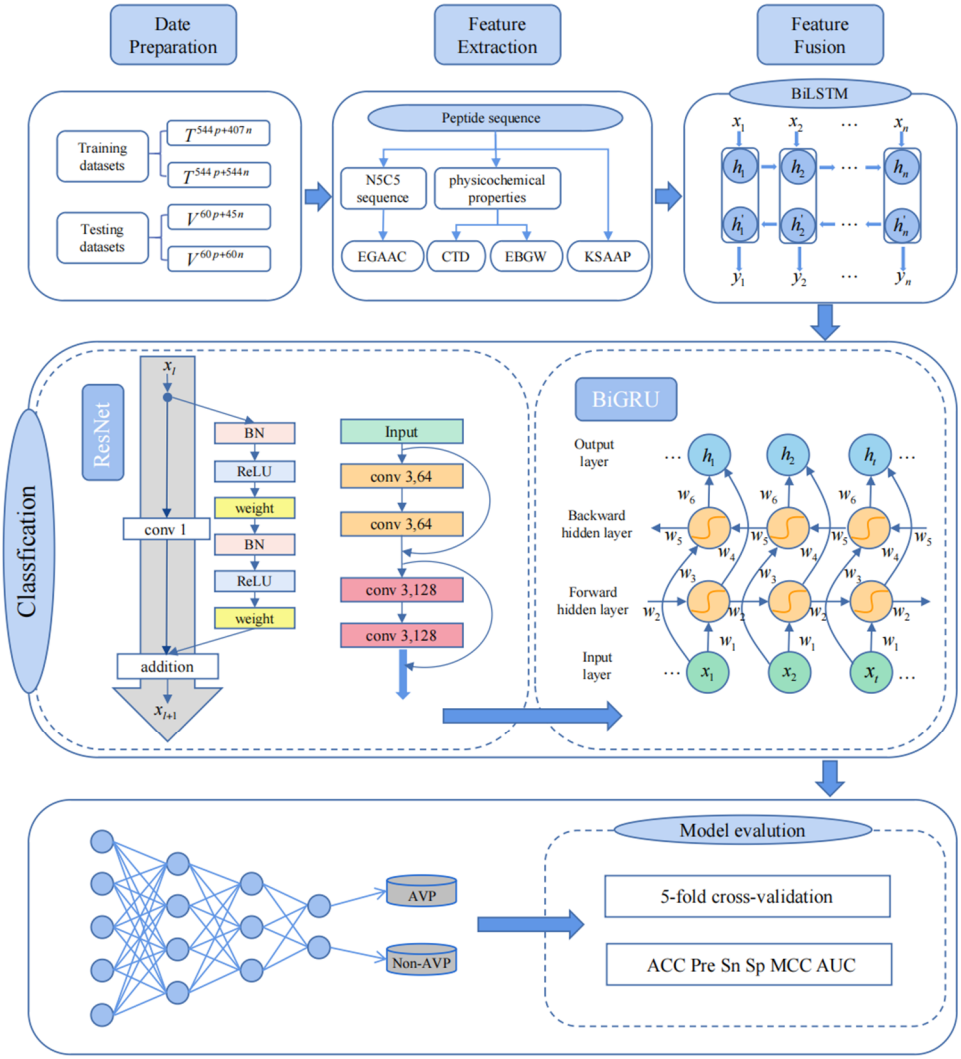


**Figure 1.** The flowchart of the iAVP-ResBi model.

## *2.2. Feature extraction*

Currently, the following feature extraction methods have been used in the existing studies targeting identification of antiviral peptides: amino acid composition (AAC) [20–24], pseudo amino

acid composition (PseAAC) [25–28], physicochemical property-based feature [27–30], amphiphilic pseudo amino acid composition (Am-PseAAC) [24,31], dipeptide composition (DPC) [24,25], g-gap dipeptide composition (GDC) [24,29], one-hot coding [26], composition of k-spaced amino acid group pairs (CKSAAGP) [27,28], binary profile (BP) [31–33], word2vec encoding [33] and BLOSUM62 encoding [33].

In order to broaden the sources of features information and overcome the uniqueness and one-sidedness of extracted features, it is necessary to delve as deeply as possible into the internal relationships expressed in the data. In general, deep and efficient features can provide a strong basis for effectively improving classification accuracy [38]. In this paper, the KSAAP, EGAAC based on the N5C5 strategy, EBGW and CTD are used for extracting features. Compared with K-mer, KSAAP can describe localized fragment sequence information for amino acid pairs separated by *k* residues. Compared with GAAC, EGAAC can reflect both components and physicochemical properties, while reflecting the localized nature of the protein sequence. It is now well established that the most important influence on protein folding is the uniqueness of amino acid residues, and our choice of EBGW can reflect the distribution of residues with the same unique characteristics, as well as depict the nature of the protein sequence. CTD can account for differences in properties in the presence of different patterns of amino acid distribution. These feature extraction methods have been widely applied to the identification of other therapeutic peptides [39,40] and post-translational modification sites [41–43].

#### 2.2.1. K-spaced amino acid pairs

K-spaced amino acid pairing (KSAAP) is an efficient feature extraction strategy that can be used to highlight and identify motifs in protein fragments and sequences [44–46]. KSAAP can be used to identify flexible and rigid regions of proteins and has been successfully applied to identify a variety of post-translational modification sites [42,43]. This coding provides a valuable descriptor for the accurate classification of AVPs and non-AVPs on the basis of short-term interactions of residues in the sequence. The detailed procedure of KSAAP is described as follows [43,46]. For a peptide sequence fragment, it calculates the frequency of the occurrence of amino acid pairs separated by *k* residues [47]. KSAAP is described as follows:

$$F_0 = \left( \frac{M_{AA}}{N_0}, \frac{M_{AC}}{N_0}, \frac{M_{AD}}{N_0}, \ldots, \frac{M_{YY}}{N_0} \right)_{400}, \tag{1}$$

$$F_1 = \left( \frac{M_{AxA}}{N_1}, \frac{M_{AxC}}{N_1}, \frac{M_{AxD}}{N_1}, \ldots, \frac{M_{YxY}}{N_1} \right)_{400}, \tag{2}$$

$$F_2 = \left( \frac{M_{AxxA}}{N_2}, \frac{M_{AxxC}}{N_2}, \frac{M_{AxxD}}{N_2}, \ldots, \frac{M_{YxxY}}{N_2} \right)_{400}, \tag{3}$$

$$F = F_0 \bigcup F_1 \bigcup F_2. \tag{4}$$

Here, we choose $k = 0$, 1 and 2. For example, $M_{AxxA}, M_{AxxC}, M_{AxxD}, \cdots, M_{YxxY}$ represent the number for 2-

spaced amino acid pairs. $N_k = L - k - 1, k = 0,1,2$, where $L$ represents the length of the peptide sequence. For each value of $k$, $F_k$ represents the features for $k$-spaced amino acid pairs. All protein sequences studied in this paper are composed of 20 natural amino acids, that is, no virtual amino acids, so the dimension of each feature space is $20 \times 20 = 400$. $F_0$, $F_1$ and $F_2$ are calculated as shown in Eqs (1)–(3), respectively. $F_0$, $F_1$ and $F_2$ are combined to get the final $400 \times 3 = 1200$-dimensional feature space $F$, as shown in Eq (4).

### 2.2.2. Enhanced grouped amino acid composition

The enhanced grouped amino acid composition (EGAAC) algorithm is an improved approach to the GAAC algorithm put forward by Chen et al. [48]. The EGAAC algorithm converts character information of peptide sequences into digital vectors. It has been successfully used to predict lysine in succinate sites [49]. As the peptide sequences used in this study vary in length, with the shortest sequence consisting of six amino acid residues, EGAAC is used to extract isometric features from the N5C5 sequence, because the N and C terminal residues are important for the structure and function of bioactive peptides [50,51]. The N5C5 sequence is constructed by truncating five amino acids from the N-terminal and C-terminal parts of the peptide sequence, and combining them into a new sequence with ten amino acid residues. Compared to fill a short sequence with virtual amino acid 'X' to the longest sequence, the N5C5 sequence strategy can avoid feature redundancy. At the same time, this method better preserves useful information, as the N-terminal and C-terminal parts of the sequence contain more essential information.

According to the five physical and chemical properties of amino acids, Lee et al. [52] classify 20 types of standard amino acids into the following five categories:

Aliphatic group: $g_1$ = {G, A,V, L, M, I}; Aromatic group: $g_2$ = {F, Y, W}; Positive charge group: $g_3$ = {K, R, H}; Negative charge group: $g_4$ = {D, E}; Uncharged group: $g_5$ = {S, T, C, P, N, Q}.

EGAAC scans along the N5C5 sequence with a fixed size window $n$. The calculation formula is as follows:

$$G(g,n) = \frac{N(g,n)}{N(n)}, g \in \{g_1, g_2, g_3, g_4, g_5\}, \tag{5}$$

where $N(g,n)$ represents the number of amino acids belonging to the $g$-th group in the window $n$, and $N(n)$ represents the length of the window. The value of window length is from 1 to $L$, since EGAAC is analyzed on N5C5 sequence, that is $L$ = 10. In this paper, $n$ is set to 5 by default, and we finally obtain a $(L - n + 1) \times 5 = 30$-dimensional vector for each peptide sequence.

### 2.2.3. Encoding based on grouped weight

EBGW is a coding scheme [53,54] proposed by Zhang et al. [55], which characterizes sequences on the basis of the physical and chemical properties of the amino acids. On the basis of their physical and chemical properties, the 20 amino acids are divided into four groups:

Neutral and non-polarity residue group: $G_1$ = {A, F, G, I, L, M, P, V, W}; Neutral and polarity group: $G_2$ = {C, N, Q, S, T, Y}; Acidic group: $G_3$ = {D, E}; Basic group: $G_4$ = {H, K, R}.

The above four sets are combined to get three combinations, $G_1+G_2 vs. G_3+G_4$, $G_1+G_3 vs. G_2+G_4$, $G_1+G_4 vs. G_2+G_3$. Given a peptide sequence $P=p_1p_2\cdots p_L$ with length *n*, it can be transformed into three binary sequences as follows:

$$H_1(p_i)=\begin{cases}1, p_i\in G_1+G_2\\ 0, p_i\in G_3+G_4\end{cases},$$
$$H_2(p_i)=\begin{cases}1, p_i\in G_1+G_3\\ 0, p_i\in G_2+G_4\end{cases}, \quad (6)$$
$$H_3(p_i)=\begin{cases}1, p_i\in G_1+G_4\\ 0, p_i\in G_2+G_3\end{cases},$$

where $p_i(i=1,2,\cdots,L)$ belongs to any one of 20 amino acids.

Then, each sequence is cut into *J* subsequence fragments. For instance, for $H_1$, the feature of $j$-th subsequence is defined as the following:

$$X_1(j)=\frac{Sum(j)}{D(j)}, \quad (7)$$

where $Sum(j)$ is the number of 1 s in the *j*th subsequence, $D(j)=\text{int}(j\times L/J)$ refers to the length of the *j*th subsequence, $\text{int}(\cdot)$ is a function that rounds a numerical value down to the nearest integer. The EBGW scheme defines a peptide sequence as the $3\times J$ dimension vector. Here, *J* is selected as 1, 2, 3, 4 and 5. Therefore, the dimension of the EBGW-based feature vector is $3\times 15=45$.

### 2.2.4. Composition, transition and distribution

The CTD scheme is originally proposed by Dubchak et al. [56] to predict protein folding classes. This feature extraction method can be used to describe the overall amino acid composition of each peptide sequence [57]. All amino acids are classified into three categories: polar, neutral and hydrophobic. Each amino acid is then coded as 1, 2 or 3, depending on the category to which it belongs. Composition (C) describes the overall percentage composition of 20 natural amino acids, which is defined as:

$$C=\frac{n_s}{L}, s=1,2,3, \quad (8)$$

where $n_s$ is the number of the *s*th class in the sequence and *L* is the length of the peptide sequence.

The transition (T) represents the percentage frequency of one type of amino acid followed by another type of amino acid, which can be described as follows:

$$T=\frac{n_{xy}+n_{yx}}{L-1}, \text{xy}=[12],[13],[23], \quad (9)$$

where $n_{xy}$ and $n_{yx}$ are the number of dipeptides encoded as 'xy' and 'yx', respectively.

Distribution (D) is to calculate the positions of the first, 25, 50, 75 and 100% of each type of 20 natural amino acids, and the descriptor $E_i$ is defined as:

$$
\begin{aligned}
&E_i1D_x = \frac{P_1}{L}; \quad E_i25D_x = \frac{P_{25}}{L}; \quad E_i50D_x = \frac{P_{50}}{L}; \quad E_i75D_x = \frac{P_{75}}{L}; \\
&E_i100D_x = \frac{P_{100}}{L}(i = 1,2,\cdots,7; \quad x = 1,2,3),
\end{aligned} \tag{10}
$$

where $P_1, P_{25}, P_{50}, P_{75}$ and $P_{100}$ respectively measure the position of the first residue, and the occurrence rate of $X$ at 25, 50, 75 and 100%, respectively.

According to the seven physical and chemical properties of hydrophobicity, van der Waals volume, polarity, polarizability, charge, secondary structure and solvent proximity [58], the dimension of the feature vector based on CTD is finally $(3+3+3\times5)\times7 = 147$.

### *2.3. Feature fusion*

Feature fusion is an effective strategy for processing multiple features. It allows redundant information between different features to be eliminated and discrete information to be merged. Here, BiLSTM is used to combine the four feature types mentioned above.

LSTM is based on the principle of recurrent neural network (RNN) [59], where each unit is connected in turn to constitute a directional cycle. After this connection, the internal state of the network has been well constructed. At the same time, LSTM can avoid vanishing gradients, exploding gradients and poor ability to rely on long-range information. The LSTM units dynamically adapt to the length of the studied sequence. Each LSTM unit consists of an input gate, a forgetting gate and an output gate [60]. The formula can be expressed as follows:

$$
\begin{cases}
f_t = \sigma\left(W_f \cdot \left[h_{t-1}, x_t\right] + b_f\right) \\
i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_t\right] + b_i\right) \\
\tilde{C}_t = \tanh\left(W_C \cdot \left[h_{t-1}, x_t\right] + b_C\right) \\
C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \\
o_t = \sigma\left(W_o . \left[h_{t-1}, x_t\right] + b_o\right) \\
h_t = o_t \odot \tanh\left(C_t\right)
\end{cases}, \tag{11}
$$

where $f_t, i_t, C_t, o_t, h_t$ represent the forgetting gate, input gate, cell state, output gate and hidden unit state at time $t$, respectively. $\tilde{C}_t$ denotes the temporary state (candidate cell information). $W_f, W_i, W_C, W_o$ are weight matrices, $b_f, b_i, b_C, b_o$ are bias vectors, $[\,]$ means that two vectors are connected and $\odot$ means pointwise multiplication.

In this paper, BiLSTM is a combination of forward LSTM and backward LSTM. BiLSTM performs reverse processing on the input sequence information and recalculates it according to the long-term and short-term memory algorithms [61]. Fixed-length time-step outputs in both directions are obtained and the outputs from both directions are combined into a feature vector.

### *2.4. Deep learning algorithm*

In order to more accurately identify AVPs, we develop a framework of deep learning based on the residual network [62] and the bidirectional gate recurrent unit [63], which is an improvement on the framework proposed by Zhang et al. [64]. As the number of layers in the depth of the network increases, the identification accuracy reaches a saturated state then produces a rapid degradation. And residual neural networks make these layers to fit the residual mapping, rather than having each stacked layer directly fit the desired underlying mapping. He et al. [62] have showed that BN and ReLU are full pre-activation, having smaller classification error than the original residual unit. The BN layer in front of the Relu function can bring regularization effect, reduce over-fitting, and thus get higher accuracy. Afterwards we choose the bidirectional gated recurrent neural network to better capture dependencies with large step distances in the sequence. It controls the flow of information through a door that can be learned.

In this paper, first, the normalization process of the batch normalization layers (BN layer) and Relu activation function are carried out. Then, in the convolution processing, each part of the convolution is composed of two convolution layers. In the residual unit, the input and output information are added by means of conv shortcut, and two residual blocks are connected at the same time. The bidirectional gate recurrent unit includes three BiGRU layers, three dropout layers, a flatten layer, two dense layers, and a dense layer as output layer with softmax activation function. The softmax function first converts logits into a probability distribution and then select the node with the highest probability as our prediction class. The deep learning framework is operated in Python 3.8 and TensorFlow 2.5.0 under PyCharm, and the operating system is 64-bit Windows 10.

A residual network is a kind of hop-connected network that skips intermediate layers and passes previous activation values directly to the next network. In this way, the problems of gradient loss and bursting are effectively mitigated and the depth of the network to be learnt is significantly improved. At the same time, the phenomenon of information missing has been basically solved. The general structure of the network is that information is fed sequentially to each layer for processing. The residual unit can directly transfer the input $x$ to the output as the preliminary result by means of shortcut connection, and the output result is $H(x) = F(x) + x$, where $H(x) - x$ is the residual. This means that the cell inputs are added directly to the cell outputs and then activated [65], as shown in Figure 2.

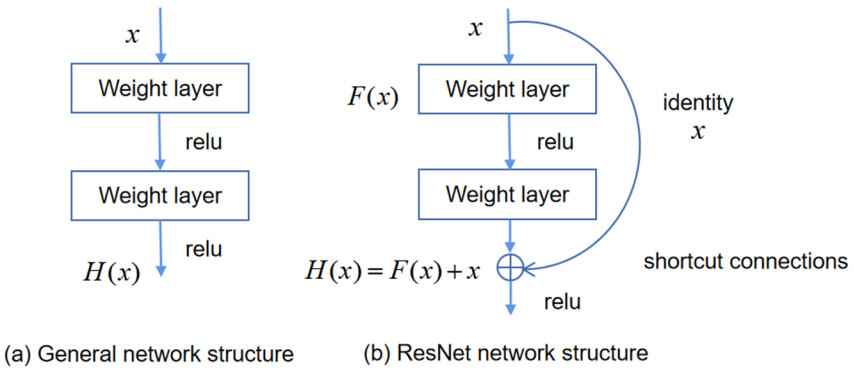


**Figure 2.** The comparison of general network structure and ResNet network structure.

The more layers in the network, the more features can be extracted at different levels. However, a high number of layers in the network can saturate and even reduce the accuracy of the training set. The key part of the residual structure shortcut connection is equivalent of a simple identity mapping. In addition, the whole network can still be trained by end-to-end back propagation. Each residual unit can be expressed by the following formulas:

$$y_l = h(x_l) + F(x_l, W_l), \tag{12}$$

$$x_{l+1} = f(y_l), \tag{13}$$

where $F$ is the residual function, which is a stack of two 1-dimensional (1D) convolutional layers with kernel_size = 3. $W_l = \{W_{l,k} \mid 1 \le k \le K\}$ is a set of biases associated with the *l*-th residual unit. The skip connection $h(x_l)$ represents the transform of 1D convolution with kernel_size = 1 and $f$ is the Relu activation function [65]. In this paper, we use the improved residual neural network, that is, the BN layer and Relu activation function are calculated first, and then the weight layer is used to calculate. The feature information can be directly transferred from one unit to another, which maintains much of the integrity of the information.

Our deep learning algorithm integrates residual unit and bidirectional gated recurrent unit. Gated recurrent neural network can solve the problem of vanishing gradients [66]. It has been widely used to process data. The traditional GRU structure consists of two parts: reset gate and update gate. The most critical step of GRU is to update the memory stage. At this stage, forgetting and remembering occur simultaneously, and update gates are used to control the extent to which information from the previous state is incorporated into the current state [67]. Update the GRU cell at each time step *t* by applying the following equation:

$$\begin{cases} r_t = \sigma\left(W_r \cdot x_t + W_r \cdot h_{(t-1)} + b_r\right) \\ \hat{h}_t = \tanh\left(W_h \cdot x_t + W_h \cdot \left(r_t \odot h_{(t-1)}\right) + b_h\right) \\ z_t = \sigma\left(W_z \cdot x_t + W_z \cdot h_{(t-1)} + b_z\right) \\ h_t = \left(1 - z_t\right) \odot h_{(t-1)} + z_t \odot \hat{h}_t \end{cases}, \tag{14}$$

where $z_t$ denotes the update gate, $r_t$ denotes the reset gate, $\hat{h}_t$ denotes the candidate status, $x_t$ is the input vector, $h_t$ is the output vector and $\odot$ means pointwise multiplication.

BiGRU is an improvement on the GRU and helps to improve the accuracy of prediction. It consists of two GRUs: a forward GRU model that takes forward input, and a reverse GRU model that learns backward input. The expression is as follows:

$$\vec{h}_t = GRU\left(x_t, \vec{h}_{t-1}\right), \ \overleftarrow{h}_t = GRU\left(x_t, \overleftarrow{h}_{t-1}\right), \tag{15}$$

$$y_t = [\vec{h}_t, \overleftarrow{h}_t], \tag{16}$$

where $\vec{h}_t$ is the forward output, $\overleftarrow{h}_t$ is the backward output, [] means that two vectors are connected and $y_t$ is the output of the model.

### *2.5. Evaluation metrics*

It is not uncommon for overfitting to occur during training, which means the model can match the training data well, but does not give a good result for data outside the training set. If test data is used to adjust the model parameters at this point, the accuracy of the final evaluation results will be affected. It is now common practice to use cross-validation and independent dataset tests to evaluate the training and generalization ability of constructed model [68]. In this paper, we choose 5-fold cross-validation and independent validation to estimate the prediction performance of this model. Meanwhile, some valid evaluation indexes are chosen to verify the feasibility of the model. They are sensitivity (Sn), specificity (Sp), accuracy (ACC), precision (Pre), recall (Rec), the receiver operating characteristic curve (ROC), precision-recall curve (PRC) and Matthew's correlation coefficient (MCC) [69–72]. They are calculated as follows:

$$\begin{cases} Sn \text{ or } Rec = \dfrac{TP}{TP + FN} \\ Sp = \dfrac{TN}{TN + FP} \\ ACC = \dfrac{TP + TN}{TP + TN + FP + FN} \\ Pre = \dfrac{TP}{TP + FP} \\ MCC = \dfrac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FN)(TP + FP)(TN + FP)(TN + FN)}} \end{cases}, \tag{17}$$

where *TP*, *TN*, *FP* and *FN* represent the number of true positive, true negative, false positive and false negative, respectively. The ROC curve shows 1-specificity on the *x* axis and sensitivity on the *y* axis, and the area under the curve is called auROC (AUC). The PR curve shows the recall on the *x* axis and the precision on the *y* axis, and the area under the curve is called auPRC. The area under the two curves ranges from 0 to 1, which is used to evaluate the effectiveness of the model [73]. Obviously, the larger the value of auROC and auPRC, the better the performance of the predictor.

## 3. Results and discussion

### *3.1. Sequence analysis*

To analyze which AVPs residues are preferred at which positions, the frequencies of occurrence of the N-terminal and C-terminal residues are examined using two sample sequence logos. Two Sample Logo is a web-based tool that can compare two groups of multi-sequences, show the amino acid composition at a specific position in the sequence, and get their statistically significant differences. The shortest length of AVPs is 6 in this study. Since AVP activity is concentrated in the N-terminal and C-terminal regions of the sequence, we study 1 to 5 N-terminal and C-terminal amino acids by sequence composition preference analysis. We submit the positive and negative samples for 5N-terminal and 5C-terminal amino acids from $T^{544p+407n}$ and $V^{60p+45n}$ datasets to the sample logo online server (http://www.twosamplelogo.org/) [74] to generate the two sample sequence logos. The logos

are scaled according to their statistical significance threshold of $p<0.05$ by Welch's t-test. The upper portion (enriched) is represented by positive AVPs, while lower portion (depleted) is represented by negative AVPs (non-AVPs), as shown in Figure 3.

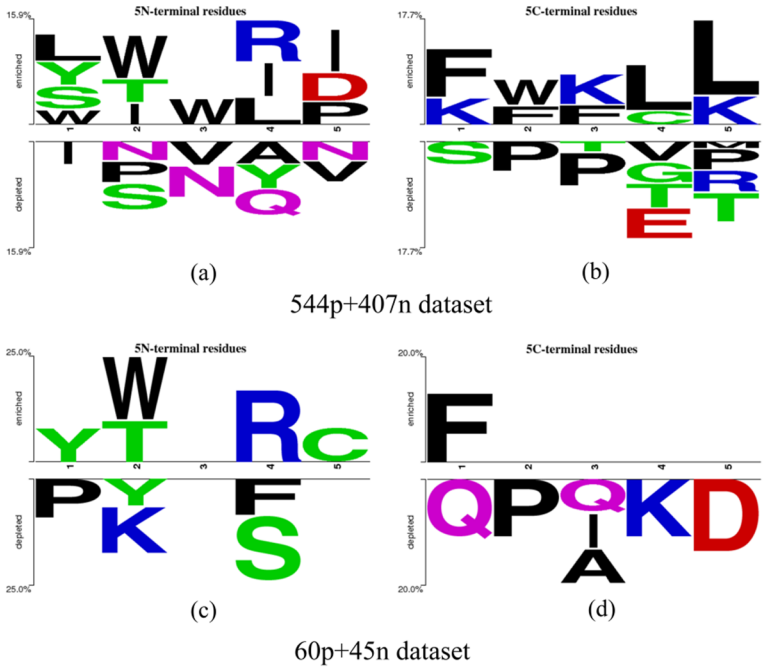


**Figure 3.** Sequence logo representation of positive and negative AVPs at 5N-terminal and 5C-terminal.

Figure 3(a),(b) expresses two sample logos for the $T^{544p+407n}$ dataset, Figure 3(c),(d) expresses two sample logos for the $V^{60p+45n}$ dataset. From Figure 3, in the $T^{544p+407n}$ dataset, we can see that positive and negative AVPs are significantly different. Positional analysis of 5 N-terminal residues showed that leucine (L) at positions 1 and 4, tryptophan (W) at positions 2 and 3, threonine (T) at position 1, arginine (R) at positions 4 and aspartate (D) at positions 5 are significantly overrepresented compared with other amino acids, while asparagine (N) at positions 2, 3 and 5, valine (V) at positions 3 and 5, alanine (A) and glutamine (Q) at position 4 are significantly underrepresented. In addition, tyrosine (Y) at position 4 is overrepresented, while Y at position 4 is underrepresented, serine (S) at position 1 is overrepresented, while S at position 2 is underrepresented, isoleucine (I) at positions 2, 4 and 5 are overrepresented while I at position 1 is underrepresented and proline (P) at position 5 is overrepresented, while P at position 2 is underrepresented. These results suggest that positive and negative AVPs are significantly different.

### *3.2. Identification performance of our model*

In this paper, we propose a model named iAVP-ResBi. Firstly, we select KSAAP to extract short-range interaction information between amino acid pairs considering the need to extract features from multiple perspectives. Due to the important information of the sequence being mostly concentrated in the N-terminal and C-terminal, we extract 30 EGAAC features based on N5C5 sequence. The different

feature parameters affect not only the size of the feature extraction, but also the subsequent predictive ability of the model. Choosing appropriate parameters is a key step in feature extraction. Selecting a parameter that is too small will lead to incomplete extracted information, while a parameter that is too large may lead to feature redundancy or even dimension disaster. In order to ensure the integrity of information, we choose the step size from 1 to 5 for EBGW, and combine the feature information of five parts. Considering the physicochemical properties of protein, we convert the peptide sequence into a numerical sequence, and extract the features by CTD. Then, the 1422 features are fused by BiLSTM, and we choose the parameter as 400 to get 800-dimensional features. Finally, we design a deep learning algorithm composed of improved residual block and a bidirectional gated recurrent unit for classification, and better experimental results are obtained.

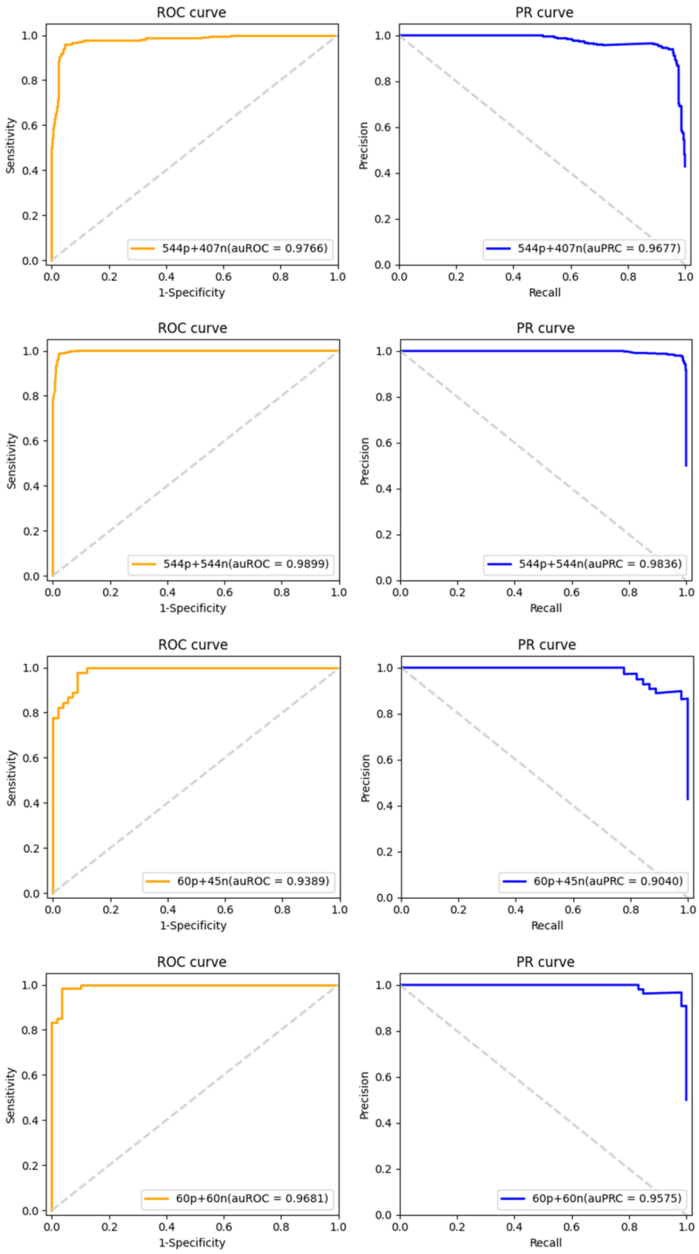


**Figure 4.** ROC curves and PR curves of $T^{544p+407n}$, $T^{544p+544n}$, $V^{60p+45n}$ and $V^{60p+60n}$ datasets.

From our model we can see that, the ACC values of the four datasets reaches 95.07, 98.07, 94.29 and 97.50%, which is a significant improvement. At the same time, the values of Sn, Sp, MCC and AUC of the four datasets have been improved to some extent. More specific information of identification results is listed in Table 1. The ROC curve and PR curve for the $T^{544p+407n}$, $T^{544p+544n}$, $V^{60p+45n}$ and $V^{60p+60n}$ datasets are plotted in Figure 4.

**Table 1.** The identification performance of iAVP-ResBi on the four datasets.

| Datasets | ACC (%) | Pre (%) | Sn (%) | Sp (%) | MCC | AUC |
|---|---|---|---|---|---|---|
| $T^{544p+407n}$ | 95.07 | 95.92 | 95.40 | 94.63 | 0.8998 | 0.9766 |
| $T^{544p+544n}$ | 98.07 | 98.65 | 97.43 | 98.71 | 0.9899 | 0.9899 |
| $V^{60p+45n}$ | 94.29 | 97.50 | 91.67 | 97.78 | 0.8962 | 0.9389 |
| $V^{60p+60n}$ | 97.50 | 98.18 | 96.67 | 98.33 | 0.9505 | 0.9681 |

**Table 2**. Performance comparison of different feature extraction methods.

| Datasets | Feature extraction | 5-fold cross-validation | | | | | |
|---|---|---|---|---|---|---|---|
| | | ACC (%) | Pre (%) | Sn (%) | Sp (%) | MCC | AUC |
| $T^{544p+407n}$ | EBGW | 82.77 | 81.98 | 91.18 | 71.59 | 0.6497 | 0.8903 |
| | EGAAC | 82.25 | 83.71 | 85.47 | 77.97 | 0.6372 | 0.8720 |
| | CTD | 91.60 | 91.76 | 93.93 | 88.50 | 0.8283 | 0.9545 |
| | KSAAP | 93.38 | 93.90 | 94.67 | 91.67 | 0.8659 | 0.9713 |
| | ALL | **95.07** | **95.92** | **95.40** | **94.63** | **0.8998** | **0.9766** |
| $T^{544p+544n}$ | EBGW | 85.40 | 85.09 | 86.04 | 84.76 | 0.7099 | 0.9278 |
| | EGAAC | 84.11 | 84.98 | 83.10 | 85.13 | 0.6826 | 0.8960 |
| | CTD | 94.03 | 94.41 | 93.57 | 94.49 | 0.8809 | 0.9776 |
| | KSAAP | 96.60 | 97.01 | 96.15 | 97.06 | 0.9325 | 0.9833 |
| | ALL | **98.07** | **98.65** | **97.43** | **98.71** | **0.9899** | **0.9899** |
| Datasets | Feature extraction | Independent validation | | | | | |
| | | ACC (%) | Pre (%) | Sn (%) | Sp (%) | MCC | AUC |
| $V^{60p+45n}$ | EBGW | 75.24 | 74.21 | 83.33 | 64.44 | 0.5070 | 0.8019 |
| | EGAAC | 85.71 | 86.13 | 90.00 | 80.00 | 0.7144 | 0.9148 |
| | CTD | 87.62 | 88.33 | 90.00 | 84.44 | 0.7536 | 0.9148 |
| | KSAAP | 83.81 | 84.24 | 93.33 | 71.11 | 0.6276 | 0.8870 |
| | ALL | **94.29** | **97.50** | **91.67** | **97.78** | **0.8962** | **0.9389** |
| $V^{60p+60n}$ | EBGW | 81.67 | 86.75 | 78.33 | 85.00 | 0.6396 | 0.8792 |
| | EGAAC | 86.67 | 88.33 | 88.33 | 84.44 | 0.7278 | 0.9241 |
| | CTD | 89.17 | 89.23 | 90.00 | 88.33 | 0.7847 | 0.9139 |
| | KSAAP | 93.33 | 95.00 | 90.00 | 96.67 | 0.8707 | 0.9611 |
| | ALL | **97.50** | **98.18** | **96.67** | **98.33** | **0.9505** | **0.9681** |

### *3.3. Analysis of different feature extraction approach*

In model recognition, feature extraction is the foundation, and the choice of its method has a vital impact on the prediction effect. In this work, we select four feature extraction methods including KSAAP, EGAAC, EBGW and CTD. These methods extract important information of peptide sequence from different perspectives. The single feature method is compared with the combined feature method and the result is shown in Table 2. Clearly, the combined features show better performance than any other single feature. Thus, the information obtained from different methods helps to improve the predictive power of the model.

### *3.4. Analysis of feature fusion approach*

The main purpose of feature fusion is to combine features from multiple sources into a better feature representation to improve model performance. We choose to use the deep learning algorithm BiLSTM to fuse the original features. Through BiLSTM processing, the features we obtained have been improved in importance and relevance. To illustrate the necessity of feature fusion, we compare the features before and after fusion, the experimental results are shown in Figure 5. Evidently, the ACC of the four datasets is improved after using the feature fusion method. This means that the classification effect of the whole model is greatly improved. Therefore, it is essential to use the feature fusion method to process features.

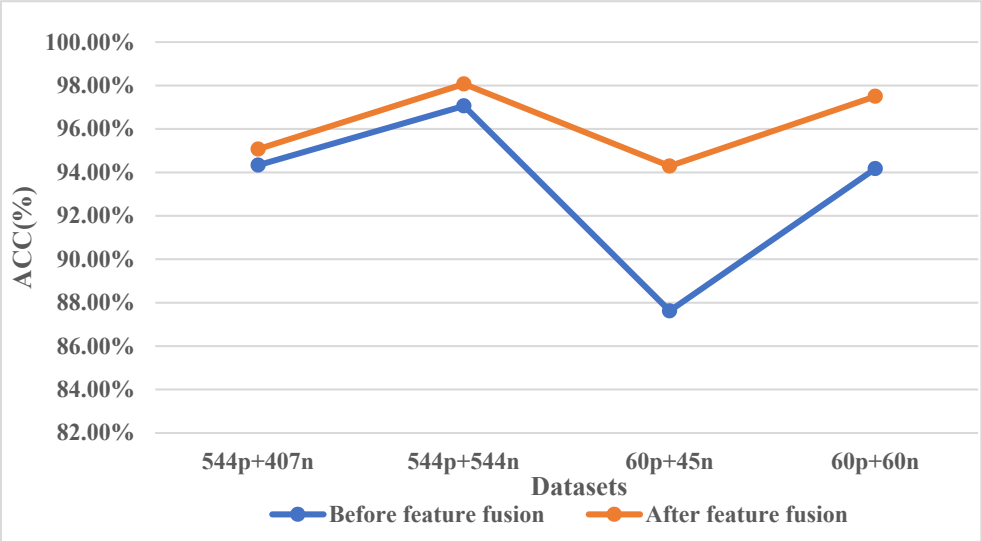


**Figure 5.** The ACC comparison of four datasets before and after feature fusion.

### *3.5. Performance comparison with different classifiers*

The most important part of the binary classification problem is the construction of the classifier, which directly affects the final test result. In this paper, we design a deep learning algorithm named ResNet + BiGRU. The combination of improved residual neural network (ResNet) and BiGRU effectively improves the identification accuracy of the model. Our ResNet used is pre-activation for

the full BN layer and Relu activation function based on the original residual network.

**Table 3**. Performance comparison of different classifiers with cross-validation.

| Datasets | Classifier | 5-fold cross-validation | | | | | |
|---|---|---|---|---|---|---|---|
| | | ACC (%) | Pre (%) | Sn (%) | Sp (%) | MCC | AUC |
| $T^{544p+407n}$ | SVM | 80.86 | 85.40 | 80.34 | 81.57 | 0.6162 | 0.8965 |
| | XGboost | 80.44 | 82.48 | 83.64 | 76.17 | 0.6018 | 0.8883 |
| | GaussianNB | 74.24 | 79.73 | 73.73 | 74.92 | 0.4831 | 0.8176 |
| | LR | 76.97 | 78.87 | 81.62 | 70.77 | 0.5278 | 0.8177 |
| | ANN | 80.33 | 82.48 | 83.46 | 76.17 | 0.5988 | 0.8777 |
| | DNN | 84.05 | 86.38 | 88.07 | 78.71 | 0.6615 | 0.8712 |
| | CNN | 85.11 | 86.64 | 87.15 | 82.38 | 0.6970 | 0.8778 |
| | BiGRU | 87.21 | 88.89 | 89.36 | 84.36 | 0.7368 | 0.8933 |
| | Original ResNet | 94.02 | 94.01 | 95.89 | 91.92 | 0.8780 | 0.9778 |
| | ResNet | 94.54 | 94.27 | 96.33 | 92.17 | 0.8895 | 0.9804 |
| | ResNet + BiLSTM | 92.13 | 92.59 | 94.48 | 89.01 | 0.8386 | 0.9658 |
| | ResNet + BiGRU | **95.07** | **95.92** | **95.40** | **94.63** | **0.8998** | **0.9766** |
| $T^{544p+544n}$ | SVM | 89.25 | 92.94 | 84.93 | 93.56 | 0.7884 | 0.9549 |
| | XGboost | 86.86 | 89.25 | 83.82 | 89.89 | 0.7386 | 0.9360 |
| | GaussianNB | 83.27 | 87.03 | 78.31 | 88.23 | 0.6694 | 0.8924 |
| | LR | 86.77 | 87.04 | 86.40 | 87.13 | 0.7357 | 0.9323 |
| | ANN | 88.88 | 89.83 | 87.69 | 90.07 | 0.7786 | 0.9434 |
| | DNN | 91.93 | 95.15 | 86.79 | 97.06 | 0.8486 | 0.9591 |
| | CNN | 92.48 | 95.28 | 88.25 | 96.70 | 0.8571 | 0.9608 |
| | BiGRU | 94.31 | 96.02 | 91.74 | 96.88 | 0.8891 | 0.9663 |
| | Original ResNet | 96.05 | 97.25 | 94.68 | 97.43 | 0.9219 | 0.9856 |
| | ResNet | 95.77 | 95.73 | 95.77 | 95.77 | 0.9158 | 0.9837 |
| | ResNet + BiLSTM | 96.42 | 97.30 | 95.41 | 97.43 | 0.9292 | 0.9842 |
| | ResNet + BiGRU | **98.07** | **98.65** | **97.43** | **98.71** | **0.9899** | **0.9899** |

The results show that ResNet + BiGRU has better performance than other classification algorithms. To describe the superiority of our deep learning algorithm, we compare ResNet+BiGRU with different classifiers. More details are shown in Tables 3 and 4. In general, the deep learning algorithm has better predictive performance than the machine learning algorithm on the same dataset. In the machine learning approaches, we compare SVM, XGboost, GaussianNB, logistic regression (LR), ANN with ResNet + BiGRU. The ACC values are 80.86, 80.44, 74.24, 76.97 and 80.33% in the $T^{544p+407n}$ dataset, respectively. For $V^{60p+45n}$, the ACC values are 60.95, 61.90, 73.33, 70.48 and 71.43%, respectively. In the deep learning approaches, we compare DNN, CNN, BiGRU, original ResNet, ResNet with ResNet+BiGRU, with ACC values of 84.05, 85.11, 87.21, 94.02 and 94.54% in the $T^{544p+407n}$ dataset, respectively, and with ACC values of 85.71, 88.57, 85.71, 88.57 and 89.52% in $V^{60p+45n}$, respectively. Compared with other deep learning algorithms, ResNet+BiGRU is an obvious improvement. In addition, the experimental results also indicate that the accuracies of the original

ResNet are lower than those of ResNet with pre-activation for the full BN layer and Relu activation function for four datasets, respectively.

**Table 4**. Performance comparison of different classifiers with independent validation.

| Datasets | Classifier | Independent validation | | | | | |
|---|---|---|---|---|---|---|---|
| | | ACC (%) | Pre (%) | Sn (%) | Sp (%) | MCC | AUC |
| $V^{60p+45n}$ | SVM | 60.95 | 68.70 | 75.00 | 42.22 | 0.1828 | 0.7833 |
| | XGboost | 61.90 | 64.46 | 76.67 | 42.22 | 0.2109 | 0.6833 |
| | GaussianNB | 73.33 | 78.55 | 73.33 | 73.33 | 0.4718 | 0.8130 |
| | LR | 70.48 | 83.14 | 63.33 | 80.00 | 0.4454 | 0.7796 |
| | ANN | 71.43 | 75.83 | 81.67 | 57.78 | 0.4386 | 0.7500 |
| | DNN | 85.71 | 89.23 | 86.67 | 84.44 | 0.7085 | 0.8889 |
| | CNN | 88.57 | 90.00 | 88.33 | 88.89 | 0.7725 | 0.8926 |
| | BiGRU | 85.71 | 89.09 | 85.00 | 86.67 | 0.7156 | 0.8574 |
| | Original ResNet | 88.57 | 89.67 | 88.33 | 88.89 | 0.7843 | 0.9046 |
| | ResNet | 89.52 | 90.13 | 91.67 | 96.67 | 0.7867 | 0.9204 |
| | ResNet + BiLSTM | 92.38 | 93.33 | 93.33 | 91.11 | 0.8444 | 0.9259 |
| | ResNet + BiGRU | **94.29** | **97.50** | **91.67** | **97.78** | **0.8962** | **0.9389** |
| $V^{60p+60n}$ | SVM | 76.67 | 87.78 | 63.33 | 90.00 | 0.5621 | 0.8639 |
| | XGboost | 67.50 | 65.72 | 73.33 | 61.67 | 0.3545 | 0.7486 |
| | GaussianNB | 78.33 | 82.73 | 71.67 | 85.00 | 0.5740 | 0.8444 |
| | LR | 75.83 | 79.83 | 71.67 | 80.00 | 0.5258 | 0.8278 |
| | ANN | 71.67 | 68.10 | 81.67 | 61.67 | 0.4426 | 0.8417 |
| | DNN | 84.17 | 80.38 | 93.33 | 75.00 | 0.6947 | 0.9042 |
| | CNN | 91.67 | 93.33 | 86.67 | 96.67 | 0.8383 | 0.9653 |
| | BiGRU | 94.17 | 94.55 | 93.33 | 95.00 | 0.8836 | 0.9681 |
| | Original ResNet | 90.00 | 91.11 | 93.33 | 86.67 | 0.8064 | 0.9542 |
| | ResNet | 90.83 | 90.53 | 96.67 | 85.00 | 0.8205 | 0.9543 |
| | ResNet + BiLSTM | 93.33 | 92.50 | 96.67 | 90.00 | 0.8707 | 0.9556 |
| | ResNet + BiGRU | **97.50** | **98.18** | **96.67** | **98.33** | **0.9505** | **0.9681** |

At the same time, to demonstrate the superiority of our combination of the ResNet and BiGRU, we also choose to combine the ResNet with BiLSTM, and the experimental results are shown in Tables 3 and 4. The final identification accuracy does not reach a more satisfactory result. Therefore, we adopt the ResNet with BiGRU to improve the accuracy. The experimental results show that our classifier has better identification performance and generalization ability. The parameters of different classifiers are shown in Table 5.

### *3.6. Performance comparison with different models*

We summarize several currently available models for identifying AVPs. To illustrate more visually the advantages of our constructed model iAVP-ResBi, seven models, namely AVPpred [20],

the model proposed by Chang et al. [21], AntiVPP 1.0 [23], Meta-iAVP [24], Firm-AVP [25], DeepAVP [26] and ENNAVIA [29], are compared using the same training dataset and test dataset. The values of four evaluation metrics for seven models are shown in Table 6.

The values in Table 6 show the predicted results of our method and the existing methods on the four datasets. Clearly, the results of our model have been greatly improved in both training set and test set. The ACC on the four datasets improved by 3.82, 2.17, 0.41 and 1.85% compared with the best model, respectively, which fully demonstrates the power of our method. The results show that our proposed iAVP-ResBi model has better predictive performance than the previous model.

**Table 5**. The parameters of different classifiers.

| Classifiers | Parameters | |
|---|---|---|
| SVM | probability = True, kernel = 'rbf', random_state = 20 | |
| XGboost | max_depth = 7, learning_rate = 0.1, n_estimators = 500 | |
| GaussianNB | priors = None, var_smoothing = 1e-09 | |
| LR | penalty = 'l2', dual = True, solver = 'liblinear', max_iter = 100 | |
| ANN | random_state = 10 | |
| DNN | Three Dense layers with 32, 16 and 8 neurons, respectively. | |
| CNN | Two Conv1D layers with filters = 32, kernel_size = 3 and two MaxPooling1D layers with pool_size = 2, strides = 1, a Flatten layer and two Dense layers with 32 and 16 neurons, respectively. | |
| BiGRU | Three layers of BiGRU with 128, 64 and 32 neurons respectively, add a dropout layer after each BiGRU layer, dropout = 0.5. | activation = "softmax", loss = categorical_crossentropy, optimizer = Adam, metrics = ["accuracy"], batch_size = 30, epochs = 30. |
| Original ResNet | Two residual blocks, each one containing Batch Normalization layer, ReLU activation function and two convolution layers with filters = 64, kernel_size = 3. | |
| ResNet | Two residual blocks, each one containing Batch Normalization layer, ReLU activation function and two convolution layers with filters = 64, kernel_size = 3. Adjust the positions of ReLU function and BN layer for pre-activation. | |
| ResNet + BiLSTM | Two residual blocks, each one containing Batch Normalization layer, ReLU activation function and two convolution layers with filters = 64, kernel_size = 3. Three layers of BiLSTM with 128, 64 and 32 neurons, respectively, add a dropout layer after each BiLSTM layer, dropout = 0.5. Add a Flatten layer and two Dense layers with 64 and 32 neurons, respectively. | |
| ResNet + BiGRU | Replace BiLSTM in ResNet + BiLSTM with BiGRU. | |

### *3.7. Universal validation of the model*

In order to further illustrate the generalizability of our constructed model, we use the dataset ENNAVIA-C created by Timmons and Hewage [29] to identify anti-coronavirus peptides. ENNAVIA-C consists of 109 peptide sequences with anti-coronavirus activity and 356 peptide sequences that have been validated to have poor or no antiviral activity. As shown in Table 7, the identification accuracy of anti-coronavirus reaches 96.99% in our constructed iAVP-ResBi model, which exceeds several

currently available models. The results show that the model we constructed is an effective recognition model with good generalization ability.

**Table 6.** Performance comparison of iAVP-ResBi and existing methods.

| Datasets | Method | 5-fold cross-validation | | | |
|---|---|---|---|---|---|
| | | ACC (%) | Sn (%) | Sp (%) | MCC |
| $T^{544p+407n}$ | AVPpred [20] | 85.00 | 82.20 | 88.20 | 0.70 |
| | Chang et al. [21] | 85.10 | 86.60 | 83.00 | 0.70 |
| | Meta-iAVP [24] | 88.20 | 89.20 | 86.90 | 0.76 |
| | DeepAVP [26] | 83.50 | 84.60 | 82.10 | 0.66 |
| | ENNAVIA [29] | 91.25 | 90.56 | 91.88 | 0.82 |
| | iAVP-ResBi | **95.07** | **95.40** | **94.63** | **0.90** |
| $T^{544p+544n}$ | AVPpred [20] | 90.00 | 89.70 | 90.30 | 0.80 |
| | Chang et al. [21] | 91.50 | 89.00 | 94.10 | 0.83 |
| | Meta-iAVP [24] | 93.20 | 89.00 | 97.40 | 0.87 |
| | DeepAVP [26] | 90.10 | 89.30 | 90.80 | 0.88 |
| | ENNAVIA [29] | 95.90 | 93.44 | 98.35 | 0.92 |
| | iAVP-ResBi | **98.07** | **97.43** | **98.71** | **0.96** |
| Datasets | Method | Independent validation | | | |
| | | ACC (%) | Sn (%) | Sp (%) | MCC |
| $V^{60p+45n}$ | AVPpred [20] | 85.70 | 88.30 | 82.20 | 0.71 |
| | Chang et al. [21] | 89.5 | 91.7 | 86.7 | 0.79 |
| | Meta-iAVP [24] | 95.20 | 96.70 | 93.20 | 0.90 |
| | Firm-AVP [25] | 92.40 | 93.30 | 91.10 | 0.84 |
| | DeepAVP [26] | 87.60 | 90.00 | 84.40 | 0.75 |
| | ENNAVIA [29] | 93.88 | 94.74 | 92.68 | 0.87 |
| | iAVP-ResBi | **94.29** | **91.67** | **97.78** | **0.90** |
| $V^{60p+60n}$ | AVPpred [20] | 92.50 | 93.30 | 91.70 | 0.85 |
| | Chang et al. [21] | 93.00 | 91.70 | 95.00 | 0.87 |
| | AntiVPP 1.0 [23] | 93.00 | 87.00 | 97.00 | 0.87 |
| | Meta-iAVP [24] | 94.90 | 91.70 | 98.30 | 0.90 |
| | DeepAVP [26] | 93.30 | 96.70 | 90.00 | 0.87 |
| | ENNAVIA [29] | 95.65 | 92.98 | 98.28 | 0.91 |
| | iAVP-ResBi | **97.50** | **96.67** | **98.33** | **0.95** |

**Table 7.** Performance comparison of iAVP-ResBi and existing models on ENNAVIA-C dataset.

| Dataset | Model | Acc (%) | Sn (%) | Sp (%) | MCC |
|---|---|---|---|---|---|
| Anti-CoV vs. Non-AVP | Pang et al's model [27] | 85.32 | 85.71 | 85.31 | 0.3050 |
| Anti-CoV vs. Non-AVP | ACP-Dnnel [75] | 95.00 | 89.40 | 100 | 0.9040 |
| ENNAVIA-C | ENNAVIA [29] | 94.95 | 91.64 | 95.96 | 0.8700 |
| ENNAVIA-C | iAVP-ResBi | **96.99** | **93.38** | **98.06** | **0.9155** |

## 4. Conclusions

As a long-term focus of researchers, antimicrobial peptides have shown great advantages in the fields of medicine and life sciences. Among them, antiviral peptides (AVPs) are an important component of antimicrobial peptides, which can not only cooperate with the components of the immune system, but also have a good therapeutic potential in drug resistance. Currently antiviral peptides are selected from databases such as HIPdb, APD3, CAMPR3 and LAMP, which are experimentally determined and time-consuming. Meanwhile, antiviral peptides have been favored by biopharmacologists in recent years due to their high pharmacological activity, strong targeting, low toxicity and more mature production technology. With the rapid development of artificial intelligence and deep learning algorithms, a large number of peptide vaccines, peptide nutraceuticals, peptide drugs and reagents will be developed and enter clinical trials in the coming years. So it is very necessary to identify AVPs accurately and efficiently. In this study, we develop a novel model called iAVP-ResBi for identification of AVPs. A single method to extract features is not comprehensive. Here we use the method of combining four features: KSAAP, EGAAC, EBGW and CTD. Then to develop a better classifier, after feature fusion through BiLSTM, the residual neural network combined with BiGRU was used to identify AVPs.

The final data shows that the model we constructed has good results. The value of ACC is 95.07, 98.07, 94.29 and 97.50% on the $T^{544p+407n}$, $T^{544p+544n}$, $V^{60p+45n}$ and $V^{60p+60n}$ datasets, respectively. As a result, the iAVP-ResBi model constructed in this paper has certain reference significance. In our future work, we will be committed to establishing the user-friendly and public web-server for our iAVP-ResBi model for the convenience of the wider research community to use. The datasets and codes are freely available at https://github.com/yunyunliang88/iAVPs-ResBi.

Our model is applicable in the identification of bioactive peptides but may not be particularly applicable in the identification of post-translational modification sites of protein or functional prediction of DNA. In addition, if the number of samples exceeds tens of thousands, our model will have some uncertainty, and at the same time, our hardware equipment may not be able to support it, so we need a high-performance computing workstation for processing.

## Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China (No.12101480), and the Fundamental Research Funds for the Central Universities (No. QTZX23002).

## Conflict of interest

The authors declare there is no conflict of interest.

## References


1. E. Domingo, Mechanisms of viral emergence, *Vet. Res.*, **41** (2010), 38. https://doi.org/10.1051/vetres/2010010
2. S. T. Nichol, J. Arikawa, Y. Kawaoka, Emerging viral diseases, *Proc. Natl. Acad. Sci.*, **97** (2000), 12411–12412. https://doi.org/10.1073/pnas.210382297
3. Q. Abid, T. Nishant, T. Himani, K. Manoj, AVPdb: a database of experimentally validated antiviral peptides targeting medically important viruses, *Nucleic Acids Res.*, **42** (2014), D1147–D1153. https://doi.org/10.1093/nar/gkt1191
4. T. Phan. Genetic diversity and evolution of SARS-CoV2, *Infect. Genet. Evol.*, **81** (2020), 104260. https://doi.org/10.1016/j.meegid.2020.104260
5. E. Sherif, A. Maha, The potential of antimicrobial peptides as an antiviral therapy against COVID-19, *ACS Pharmacol. Transl. Sci.*, **3** (2020), 780–782. https://doi.org/10.1021/acsptsci.0c00059
6. T. Uhlig, T. Kyprianou, F. G. Martinelli, C. A. Oppici, D. Heiligers, D. Hills, et al., The emergence of peptides in the pharmaceutical business: From exploration to exploitation, *EuPA Open Proteomics*, **4** (2014), 58–69. https://doi.org/10.1016/j.euprot.2014.05.003
7. L. Otvos, Peptide-based drug design: here and now, *Methods Mol. Biol.*, **494** (2008), 1–8. https://doi.org/10.1007/978-1-59745-419-3
8. R. E. W. Hancock, H. G. Sahl, Antimicrobial and host-defense peptides as new anti-infective therapeutic strategies, *Nat. Biotechnol.*, **24** (2006), 1551–1557. https://doi.org/10.1038/nbt1267
9. A, Furka, F. Sebestyén, M. Asgedom, G. Dibó, General method for rapid synthesis of multicomponent peptide mixtures, *Int. J. Pept. Protein Res.*, **37** (1991), 487–493. https://doi.org/10.1111/j.1399-3011.1991.tb00765.x
10. K. Bozovičar, T. Bratkovič, Evolving a peptide: library platforms and diversification strategies, *Int. J. Mol. Sci.*, **21** (2019), 215. https://doi.org/10.3390/ijms21010215
11. Z. Y. Lou, Y. N. Sun, Z. H. Rao, Current progress in antiviral strategies. *Trends Pharmacol. Sci.*, **35** (2014), 86–102. https://doi.org/10.1016/j.tips.2013.11.006
12. F. Yu, L. Lu, L. Y. Du, X. J. Zhu, A. K. Debnath, S. Jiang, Approaches for identification of HIV-1 entry inhibitors targeting gp41 pocket, *Viruses*, **5** (2013), 127–149. https://doi.org/10.3390/v5010127
13. C. K. McDonald, D. R. Kuritzkes, Human immunodeficiency virus type 1 protease inhibitors, *Arch. Intern. Med.*, **157** (1997), 951–959.
14. J. J. Kiser, C. Flexner, Direct-acting antiviral agents for hepatitis C virus infection, *Annu. Rev. Pharmacol. Toxicol.*, **53** (2013), 427–449. https://doi.org/10.1146/annurev-pharmtox-011112-140254
15. R. Eléonore, R. Jean-Christophe, B. Véronique, J. Corinne, P. Pierre, T, Noël, et al., Antiviral drug discovery strategy using combinatorial libraries of structurally constrained peptides, *J. Virol.*, **78** (2004), 7410–7417. https://doi.org/10.1128/JVI.78.14.7410-7417.2004
16. G. Castel, M. Chtéoui, B. Heyd, N. Tordo, Phage display of combinatorial peptide libraries: application to antiviral research, *Molecules*, **16** (2011), 3499–3518. https://doi.org/10.3390/molecules16053499
17. M. F. Chew, K. S. Poh, C. L. Poh, Peptides as therapeutic agents for dengue virus, *Int. J. Med. Sci.*, **14** (2017), 1342–1359. https://doi.org/10.7150/ijms.21875.
18. S. Saheli, Vaccination: the present and the future, *Yale J. Biol. Med.*, **84** (2011), 353–359.

19. H. B. Jiang, Y. D. Xu, L. Li, L. Y. Weng, Q. Wang, S. J. Zhang, et al., Inhibition of influenza virus replication by constrained peptides targeting nucleoprotein, *Antiviral Chem. Chemother.*, **22** (2011), 119–130. https://doi.org/10.3851/IMP1902
20. T. Nishant, Q. Abid, K. Manoj, AVPpred: collection and prediction of highly effective antiviral peptides, *Nucleic Acids Res.*, **40** (2012), W199–W204. https://doi.org/10.1093/nar/gks450
21. K. Y. Chang, J. R. Yang, Analysis and prediction of highly effective antiviral peptides based on random forests, *PloS One*, **8** (2013), e70166. https://doi.org/10.1371/journal.pone.0070166
22. M. Zare, H. Mohabatkar, F. K. Faramarzi, M. M. Beigi, M. Behbahani, Using Chou's pseudo amino acid composition and machine learning method to predict the antiviral peptides, *Open Bioinf. J.*, **9** (2015), 13–19. https://doi.org/10.2174/1875036201509010013
23. B. F. J. Lissabet, H. L. Belén, G. J. Farias, AntiVPP 1.0: A portable tool for prediction of antiviral peptides, *Comput. Biol. Med.*, **107** (2019), 127–130. https://doi.org/10.1016/j.compbiomed.2019.02.011
24. N. Schaduangrat, C. Nantasenamat, V. Prachayasittikul, W. Shoombuatong, Meta-iAVP: a sequence-based meta-predictor for improving the prediction of antiviral peptides using effective feature representation, *Int. J. Mol. Sci.*, **20** (2019), 5743. https://doi.org/10.3390/ijms20225743
25. S. C. Abu, M. R. Sarah, K. H. Kylene, B. Barney, M. W. R. Bobbie-Jo, Better understanding and prediction of antiviral peptides through primary and secondary structure feature importance, *Sci. Rep.*, **10** (2020), 19260. https://doi.org/10.1038/s41598-020-76161-8
26. J. W. Li, Y. Q. Pu, J. J. Tang, Q. Zou, F. Guo, DeepAVP: a dual-channel deep neural network for identifying variable-length antiviral peptides, *IEEE J. Biomed. Health*, **24** (2020), 3012–3019. https://doi.org/10.1109/JBHI.2020.2977091
27. Y. X. Pang, Z. Wang, J. H. Jhong, T. Y. Lee, Identifying anti-coronavirus peptides by incorporating different negative datasets and imbalanced learning strategies, *Briefings Bioinf.*, **22** (2021), 1085–1095. https://doi.org/10.1093/bib/bbaa423
28. Y. Pang, L. Yao, J. H. Jhong, Z. Wang, T. Y. Lee, AVPIden: a new scheme for identification and functional prediction of antiviral peptides based on machine learning approaches, *Brief. Bioinform.*, **22** (2021), bbab263. https://doi.org/10.1093/bib/bbab263
29. P. B. Timmons, C. M. Hewage, ENNAVIA is a novel method which employs neural networks for antiviral and anti-coronavirus activity prediction for therapeutic peptides, *Briefings Bioinf.*, **22** (2021), bbab258. https://doi.org/10.1093/bib/bbab258
30. G. Agarwal, R. Gabrani, Antiviral peptides: identification and validation, *Int. J. Pept. Res. Ther.*, **27** (2021), 149–168. https://doi.org/10.1007/s10989-020-10072-0
31. P. Charoenkwan, N. Anuwongcharoen, C. Nantasenamat, M. M. Hasan, W. Shoombuatong, In silico approaches for the prediction and analysis of antiviral peptides: a review, *Curr. Pharm. Design*, **27** (2021), 2180–2188. https://doi.org/10.2174/1381612826666201102105827
32. B. Manavalan, S. Basith, G. Lee, Comparative analysis of machine learning-based approaches for identifying therapeutic peptides targeting SARS-CoV-2, *Briefings Bioinf.*, **23** (2022), bbab412. https://doi.org/10.1093/bib/bbab412
33. H. Kurata, S. Tsukiyama, B. Manavalan, iACVP: markedly enhanced identification of anti-coronavirus peptides using a dataset-specific word2vec model, *Briefings Bioinf.*, **23** (2022), bbac265. https://doi.org/10.1093/bib/bbac265

34. M. K. Pirtskhalava, A. A. Amstrong, M. Grigolava, M. Chubinidze, E. Alimbarashvili, B. M. Vishnepolsky, DBAASP v3: database of antimicrobial/cytotoxic activity and structure of peptides as a resource for development of new therapeutics, *Nucleic. Acids Res.*, **49** (2021), D288–D297. https://doi.org/10.1093/nar/gkaa991
35. F. H. Waghu, L. Gopi, R. S. Barai, P. Ramteke, B. Nizami, S. Idicula-Thomas, CAMP: Collection of sequences and structures of antimicrobial peptides, *Nucleic. Acids Res.*, **42** (2014), D1154–D1158. https://doi.org/10.1093/nar/gkt1157
36. G. S. Wang, X. Li, Z. Wang, APD3: the antimicrobial peptide database as a tool for research and education, *Nucleic. Acids Res.*, **44** (2016), D1087–D1093. https://doi.org/10.1093/nar/gkv1278
37. S. Lata, N. K. Mishra, G. P. S. Raghava, AntiBP2: improved version of antibacterial peptide prediction, *BMC Bioinf.*, **11** (2010), S19. https://doi.org/10.1186/1471-2105-11-S1-S19
38. N. Bupi, V. K. Sangaraju, L. T. Phan, A. Lal, T. T. B. Vo, P. T. Ho, et al., An effective integrated machine learning framework for identifying severity of tomato yellow leaf curl virus and their experimental validation, *Research*, **6** (2023), 16. https://doi.org/10.34133/research.0016
39. H. Y. Shi, S. L. Zhang, Accurate prediction of anti-hypertensive peptides based on convolutional neural network and gated recurrent unit, *Interdiscip. Sci.*, **14** (2022), 879–894. https://doi.org/10.1007/s12539-022-00521-3
40. S. L. Zhang, X. J. Li, Pep-CNN: An improved convolutional neural network for predicting therapeutic peptides, *Chemometr. Intell. Lab.*, **221** (2022), 104490. https://doi.org/10.1016/j.chemolab.2022.104490
41. M. M. Hasan, Y. Zhou, X. T. Lu, J. Y. Li, J. N. Song, Z. D. Zhang, Computational identification of protein pupylation sites by using profile-based composition of k-spaced amino acid pairs, *PloS One*, **10** (2015), e0129635. https://doi.org/10.1371/journal.pone.0129635
42. Z. Ju, J. Z. Cao, Prediction of protein N-formylation using the composition of k-spaced amino acid pairs, *Anal. Biochem.*, **534** (2017), 40–45. https://doi.org/10.1016/j.ab.2017.07.011
43. Z. Ju, S. Y. Wang, Prediction of lysine formylation sites using the composition of k-spaced amino acid pairs via Chou's 5-steps rule and general pseudo components, *Genomics*, **112** (2020), 859–866. https://doi.org/10.1016/j.ygeno.2019.05.027
44. M. M. Hasan, M. S. Khatun, M. N. H. Mollah, Y. Cao, D. J. Guo, NTyroSite: Computational identification of protein nitrotyrosine sites using sequence evolutionary features, *Molecules*, **23** (2018), 1667. https://doi.org/10.3390/molecules23071667
45. H. L. Fu, Y. X. Yang, X. B. Wang, H. Wang, Y. Xu, DeepUbi: a deep learning framework for prediction of ubiquitination sites in proteins, *BMC Bioinf.*, **20** (2019), 86. https://doi.org/10.1186/s12859-019-2677-9
46. J. N. Song, Y. N. Wang, F. Y. Li, T. Akutsu, N. D. Rawlings, G. I. Webb, K. C. Chou, iProt-Sub: a comprehensive package for accurately mapping and predicting protease-specific substrates and cleavage sites, *Briefings Bioinf.*, **20** (2019), 638–658. https://doi.org/10.1093/bib/bby028
47. M. Usman, S. Khan, J. A. Lee, Afp-lse: Antifreeze proteins prediction using latent space encoding of composition of k-spaced amino acid pairs, *Sci. Rep.*, **10** (2020), 7197. https://doi.org/10.1038/s41598-020-63259-2
48. Z. Chen, P. Zhao, F. Y. Li, A. Leier, T. T. Marquez-Lago, Y. N. Wang, et al., iFeature: a python package and web server for features extraction and selection from protein and peptide sequences, *Bioinformatics*, **34** (2018), 2499–2502. https://doi.org/10.1093/bioinformatics/bty140

49. Y. Zhu, C. Z. Jia, F. Y. Li, J. N. Song, Inspector: a lysine succinylation predictor based on edited nearest-neighbor undersampling and adaptive synthetic oversampling, *Anal. Biochem.*, **593** (2020), 113592. https://doi.org/10.1016/j.ab.2020.113592
50. C. R. Chung, T. R. Kuo, L. C. Wu, T. Y. Lee, J. T. Horng, Characterization and identification of antimicrobial peptides with different functional activities, *Briefings Bioinf.*, **21** (2020), 1098–1114. https://doi.org/10.1093/bib/bbz043
51. Y. J. Wang, Q. Zhang, M. A. Sun, D. J. Guo, High-accuracy prediction of bacterial type III secreted effectors based on position specific amino acid composition profiles, *Bioinformatics*, **27** (2011), 777–784. https://doi.org/10.1093/bioinformatics/btr021
52. T. Y. Lee, Z. Q. Lin, S. J. Hsieh, N. A. Bretana, C. T. Lu, Exploiting maximal dependence decomposition to identify conserved motifs from a group of aligned signal sequences, *Bioinformatics*, **27** (2011), 1780–1787. https://doi.org/10.1093/bioinformatics/btr291
53. X. Y. Wang, B. Yu, A. J. Ma, C. Chen, B. Q. Liu, Q. Ma, Protein-protein interaction sites prediction by ensemble random forests with synthetic minority oversampling technique, *Bioinformatics*, **35** (2019), 2395–2402. https://doi.org/10.1093/bioinformatics/bty995
54. B. G. Tian, X. Wu, C. Chen, W. Y. Qiu, Q. Ma, B. Yu, Predicting protein-protein interactions by fusing various Chou's pseudo components and using wavelet denoising approach, *J. Theor. Biol.*, **462** (2019), 329–346. https://doi.org/10.1016/j.jtbi.2018.11.011
55. Z. H. Zhang, Z. H. Wang, Z. R. Zhang, Y. X. Wang, A novel method for apoptosis protein subcellular localization prediction combining encoding based on grouped weight and support vector machine, *FEBS Lett.*, **580** (2006), 6169–6174. https://doi.org/10.1016/j.febslet.2006.10.017
56. I. Dubchak, I. Muchnik, S. R. Holbrook, S. H. Kim, Prediction of protein folding class using global description of amino acid sequence, *Proc. Natl. Acad. Sci.*, **92** (1995), 8700–8704. https://doi.org/10.2307/2368330
57. Z. R. Li, H. H. Lin, L. Y. Han, L. Jiang, X. Chen, Y. Z. Chen, PROFEAT: a web server for computing structural and physicochemical features of proteins and peptides from amino acid sequence, *Nucleic. Acids Res.*, **34** (2006), W32–W37. https://doi.org/10.1093/nar/gkl305
58. H. Lv, F. Y. Dao, Z. X. Guan, H. Yang, Y. W. Li, H. Lin, Deep-Kcr: accurate detection of lysine crotonylation sites using deep learning method, *Briefings Bioinf.*, **22** (2021), bbaa255. https://doi.org/10.1093/bib/bbaa255
59. S. Hochreiter, J. Schmidhuber, Long short-term memory, *Neural Comput.*, **9** (1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735
60. A. Graves, J. Schmidhuber, Framewise phoneme classification with bidirectional LSTM and other neural network architectures, *Neural Networks*, **18** (2005), 602–610. https://doi.org/10.1016/j.neunet.2005.06.042
61. J. W. Li, Y. Q. Pu, J. J. Tang, Q. Zou, F. Guo, DeepATT: a hybrid category attention neural network for identifying functional effects of DNA sequences, *Briefings Bioinf.*, **22** (2021), bbaa159. https://doi.org/10.1093/bib/bbaa159
62. K. M. He, X. Y. Zhang, S. Q. Ren, J. Sun, Deep residual learning for image recognition, *Comput. Sci.*, (2016), 770–778. https://doi.org/10.48550/arXiv.1512.03385
63. R. Zhou, X. Q. Hu, B. Yuan, Q. W. Xu, Lithology classification system for well logging based on bidirectional gated recurrent unit, in *2021 4th International conference on artificial intelligence and big data (ICAIBD)*, (2021), 599–603. https://doi.org/10.1109/ICAIBD51990.2021.9459000

64. S. L. Zhang, Y. Y. Jing, PreVFs-RG: A deep hybrid model for identifying virulence factors based on residual block and gated recurrent unit, *IEEE/ACM Trans. Comput. Biol. Bioinf.*, **20** (2022), 1926–1934. https://doi.org/10.1109/TCBB.2022.3223038
65. K. M. He, X. Y. Zhang, S. Q. Ren, J. Sun, Identity mappings in deep residual networks, in *Computer Vision–ECCV 2016: 14th European Conference*, (2016), 630–645. https://doi.org/10.48550/arXiv.1603.05027
66. K. Cho, B. V. Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, et al., Learning phrase representations using RNN encoder-decoder for statistical machine translation, preprint, arXiv:1406.1078. https://doi.org/10.48550/arXiv.1406.1078
67. K. Abdelli, J. Y. Cho, F. Azendorf, H. Griesser, C. Tropschug, S. Pachnicke, Machine-learning-based anomaly detection in optical fiber monitoring, *J. Opt. Commun. Networking*, **14** (2022), 365–375. https://doi.org/10.1364/JOCN.451289
68. Q. H. Kha, Q. T. Ho, N. Q. K. Le, Identifying SNARE proteins using an alignment-free method based on multiscan convolutional neural network and PSSM profiles, *J. Chem. Inf. Model.*, **62** (2022), 4820–4826. https://doi.org/10.1021/acs.jcim.2c01034
69. N. Q. K. Le, T. T. D. Nguyen, Y. Y. Ou, Identifying the molecular functions of electron transport proteins using radial basis function networks and biochemical properties, *J. Mol. Graph. Model.*, **73** (2017), 166–178. https://doi.org/10.1016/j.jmgm.2017.01.003
70. W. Shoombuatong, S. Basith, T. Pitti, G. Lee, B. Manavalan, THRONE: a new approach for accurate prediction of human RNA N7-methylguanosine sites, *J. Mol. Biol.*, **434** (2022), 167549. https://doi.org/10.1016/j.jmb.2022.167549
71. L. Y. Wei, W. J. He, A. Malik, R. Su, L. Z. Cui, B. Manavalan, Computational prediction and interpretation of cell-specific replication origin sites from multiple eukaryotes by exploiting stacking framework, *Briefings Bioinf.*, **22** (2021), bbaa275. https://doi.org/10.1093/bib/bbaa275
72. M. M. Hasan, S. Tsukiyama, J. Y. Cho, H. Kurata, M. A. Alam, X. W. Liu, et al.,Deepm5C: a deep-learning-based hybrid framework for identifying human RNA N5-methylcytosine sites using a stacking strategy, *Mol. Ther.*, **30** (2022), 2856–2867. https://doi.org/10.1016/j.ymthe.2022.05.001
73. P. Charoenkwan, W. Chiangjong, C. Nantasenamat, M. M. Hasan, B. Manavalan, W. Shoombuatong, StackIL6: a stacking ensemble model for improving the prediction of IL-6 inducing peptides, *Briefings Bioinf.*, **22** (2021), bbab172. https://doi.org/10.1093/bib/bbab172
74. V. Vacic, L. M. Iakoucheva, P. Radivojac, Two Sample Logo: a graphical representation of the differences between two sets of sequence alignments, *Bioinformatics*, **22** (2006), 1536–1537. https://doi.org/10.1093/bioinformatics/btl151
75. M. Y. Liu, H. M. Liu, T. Wu, Y. X. Zhu, Y. W. Zhou, Z. R. Huang, et al., ACP-Dnnel: anti-coronavirus peptides' prediction based on deep neural network ensemble learning, *Amino Acids*, **55** (2023), 1121–1136. https://doi.org/10.1007/s00726-023-03300-6


AIMS Press