MATCH
Communications in Mathematical and in Computer Chemistry

MATCH Commun. Math. Comput. Chem. **91** (2024) 109–131
ISSN: 0340–6253
doi: 10.46793/match.91-1.109Z

AMP-EF: An Ensemble Framework of Extreme Gradient Boosting and Bidirectional Long Short-Term Memory Network for Identifying Antimicrobial Peptides

Shengli Zhang[a,b,*], **Ya Zhao**[a], **Yunyun Liang**[c]

[a] *School of Mathematics and Statistics, Xidian University, Xi'an 710071, P. R. China*

[b] *Key Laboratory of Computational Science and Application of Hainan Province, Haikou, 571158, China*

[c] *School of Science, Xi'an Polytechnic University, Xi'an 710048, P. R. China*

shengli0201@163.com

(Received June 2, 2023)

Abstract

In recent years, bacterial resistance becomes a serious problem due to the abuse of antibiotics. Antimicrobial peptides (AMPs) have rapidly emerged as the best alternative to antibiotics because of their ability to rapidly target bacteria, fungi, viruses, and cancer cells and counteract the toxins they produce. In this study, a two-branch ensemble framework is proposed to identify AMPs, which integrates extreme gradient boosting (XGBoost) and bidirectional long short-term memory network (Bi-LSTM) with attention mechanism to form a stronger model. First, one-hot coding and k-mer are used to represent the sequence features. Then, the feature vectors are input into the two base classifiers respectively to obtain two predicted values. Finally, the prediction results are obtained

*Corresponding author.

by compromise. As one of the classical machine learning methods, XGBoost has strong stability and can adapt to datasets of different sizes. Bi-LSTM recurses for each peptide from N-terminal to C-terminal and C-terminal to N-terminal, respectively. As the context information is provided, the model can make more accurate prediction. Our method achieves higher or highly comparable results across the eight independent test datasets. The ACC values of XUAMP, YADAMP, DRAMP, CAMP, LAMP, APD3, dbAMP, and DBAASP are 77.9%, 98.5%, 72.5%, 99.8%, 83.0%, 92.4%, 87.5%, and 84.6%, respectively. This shows that the two-branch ensemble structure is feasible and has strong generalization. The codes and datasets are accessible at https://github.com/z11code/AMP-EF.

1 Introduction

So far, more than 800 AMPs have been identified from bacteria, fungi, amphibians, insects, higher plants, mammals, and even humans [1]. It consists of protein-based amino acid chains, typically between 6 and 100 in length. AMPs are the first immunoactive molecules produced in organisms. They play an extremely important role in the host immune defense against pathogen invasion and are known as "natural antibacterial agents" [2]. Because of the increasing number of bacteria that are resistant to antibiotics, AMPs are seen as a viable alternative form of treatment. In theory, customized peptides treat infections, enhance immune responses, and counteract toxins produced by microbes. In addition, studies have found that AMPs also have potent killing effects on some fungi, protozoa, viruses, and cancer cells [3]. Therefore, the study and identification of AMPs have received increasing attention. It takes a lot of time and money to identify AMPs by biological experiments, but building a computational model to identify AMPs can not only reduce the cost but also improve the accuracy.

As early as 2007, researchers began to use machine learning algorithms to predict AMPs [4], with support vector machines (SVM) [5] and random forests (RF) [6, 7] being the most commonly used. In 2019, Chung et al. constructed a classifier named AMPfun that extracted features reflecting the functional activity of AMPs and used RF as the classifier [8]. In 2020, Michal Burdukiewicz et al. designed a prediction model for longer AMPs,

AmpGram, which used RF as the identification algorithm [9]. In 2020, Legana C.H. Wfingerhut et al. developed an R package with a built-in SVM classifier, called ampir [10]. In 2020, Kaveh Kavousi et al. built a prediction platform IAMPE, which involved SVM, RF, XGBoost, and other identification algorithms [11]. In recent years, with the rise of neural networks, deep learning has also been used to identify AMPs. In 2018, Daniel Veltri et al. first applied deep learning methods to the identification of AMPs. Their AMPScannerV2 model included convolutional neural network (CNN) and long short-term memory network (LSTM) [12]. In 2019, Su et al. designed a multi-scale convolutional network to identify AMPs, extracting all potential features with multiple convolution kernels of different sizes [13]. In 2020, Yan et al. also created Deep-AmPEP30 using CNN to predict short AMPs [14]. In 2022, Li et al. first introduced attention mechanism and combined it with Bi-LSTM to build the AMPlify model [15]. In 2022, Yan et al. proposed a graph attention network (GAT) called sAMPpred-GAT to identify AMPs and non-AMPs [16]. After reading articles related to the identification of AMPs, we find that existing models only use a single machine learning method or a single deep learning method to distinguish AMPs from non-AMPs. Cesar R. Garcia-Jacas et al. compared machine learning algorithms with deep learning algorithms and concluded that both have advantages and disadvantages when it comes to the identification of AMPs [17]. In summary, we can consider integrating different types of algorithms to improve the prediction accuracy of AMPs. Wang et al. have combined XGBoost with CNN to identify non-classical secreted proteins, and the results showed better performance than existing state-of-the-art models and other classical machine learning models [18]. Therefore, we can also try to apply a two-branch ensemble framework to the identification of AMPs.

Our ensemble framework, named AMP-EF, uses XGBoost and Bi-LSTM with attention mechanism as two branches of classifier, respectively. As an optimization strategy, ensemble learning can skillfully integrate the two, to obtain significantly superior performance than a single classifier. XGBoost is the best machine learning method for predicting AMPs except the above two most commonly used SVM and RF. In the condition

of using F-score feature selection, XGBoost has the highest prediction accuracy [19]. Bi-LSTM has a forward layer, a backward layer, and a connection layer. The output vector of each residue is the concatenation of vectors from two directions, containing both past and future data, which strengthens the dependence between data [20–22]. The output of Bi-LSTM is then used as the input of attention mechanism. Attention mechanism can effectively reduce dimension and avoid information loss by taking different attention to different features. In addition, in order to convert the original peptide into a mathematically tractable format, we adopt one-hot encoding and k-mer to represent sequence features, which make full use of the position and composition information of peptide sequences. The model structure is shown in Fig 1.

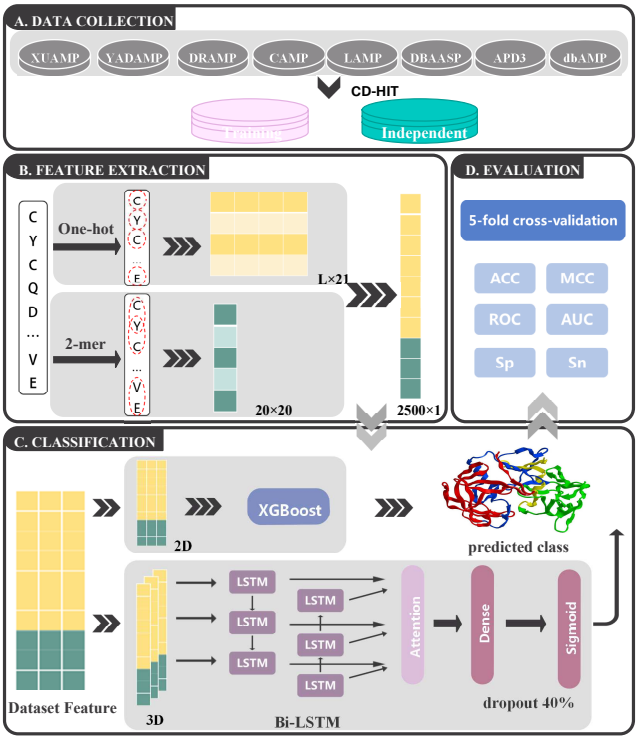


Figure 1. The architecture of AMP-EF. '2D' represents a 2-dimensional feature vector and '3D' represents a 3-dimensional feature vector.

2 Materials and methods

2.1 Dataset

To demonstrate that our model is effective for the prediction of AMPs, we adopt the datasets collected and curated by Yan et al. [16]. Yan et al. used eight independent test datasets, among which seven datasets came from Xu et al. [19], namely XUAMP [19], YADAMP [23], DRAMP [24,25], CAMP [26, 27], LAMP [28, 29], APD3 [30] and dbAMP [31]. Meanwhile, Yan et al. used the same method as Xu et al. to construct an independent test dataset called DBAASP. The details of these eight datasets can be found in Table 1. In addition, Yan et al. constructed eight datasets by randomly selecting positive and negative samples with length between 40 and 100 residues, and used CD-HIT [32] to eliminate redundancy between sequences, making the similarity between positive samples less than 90% and the similarity between negative samples less than 40%. We use them as training datasets in this paper. The XUAMP and DBAASP training datasets contain 1500 positive samples and 1500 negative samples each, and the other six training datasets contain 1000 positive samples and 1000 negative samples each [16]. In this study, we mainly construct and evaluate the model based on the XUAMP datasets, and the other seven datasets are used to verify the portability of the model. The sequence length of samples in the above datasets is different. Before prediction, we supplement all the peptide sequences to reach 100-bp by filling the letter '*O*' at the C-terminal of the sequences.

Table 1. Details of the eight independent test datasets.

Independent test datasets	Positive	Negative
XUAMP	1536	1536
YADAMP	324	324
DRAMP	1408	1408
CAMP	203	203
LAMP	1054	1054
DBAASP	178	178
APD3	494	494
dbAMP	522	522

2.2 Feature extraction

Effectively extracting sequence features is beneficial to improve the accuracy of the model. We extract features from the following two aspects: peptide sequence location information and amino acid composition. One-hot encoding can reflect the position information of the sequence and k-mer represent the composition of amino acids. And then, the $L \times 21$-dimensional feature vector $x = [x_1, x_2, \cdots, x_{2100}]$ obtained by one-hot encoding and the 400-dimensional feature vector $y = [y_1, y_2, \cdots, y_{400}]$ obtained by k-mer are concatenated to obtain the final fused feature vector $z = [x_1, x_2, \cdots, x_{2100}, y_1, y_2, \cdots, y_{400}]$.

2.2.1 One-hot encoding

Since the classifiers of machine learning and deep learning algorithms can not directly process discrete and unordered data, we first apply one-hot encoding to preprocess peptide sequences. One-hot coding, also known as one-bit efficient coding, can establish a mapping table for discrete and disordered data to make it continuous and ordered, which is a simple and effective coding method. Currently, one-hot encoding is widely used in many branches of computer science, especially machine learning, deep learning, and digital circuit design.

One-hot encoding is a representation of categorical variables as binary vectors by using N-bit state registers to encode N states, each of which has its register bit, and at any time, only one of which is valid, that is, only one of which is 1, and the rest is 0 [33]. Firstly, the classification value is mapped to the integer value. Each integer value is then represented as a binary vector with zero except for the index of the integer, which is labeled as 1 [34]. Before using one-hot encoding, the letter 'O' is needed to fill the C-terminal of a peptide sequence of varying lengths to make each sample the same length. In this way, the amino acids involved in the peptide sequence can be regarded as 21 types (20 common amino acids and 'O'), which correspond to 21-bit status registers, so each amino acid can be represented as a 21-dimensional binary vector. For example, A is represented as $[1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]$, D is represented as $[0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]$, and O is

represented as $[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1]$. For a peptide sequence of length L, the feature matrix obtained by one-hot encoding is $L \times 21$-dimensional. If a peptide sequence of 22 kinds of amino acids, each amino acid can be represented as a 23-dimensional binary vector. Same principle as above.

2.2.2 K-mer

K-mer is a coding method based on sequence composition, which constructs a number vector by calculating the frequencies of k consecutive intervals in a sequence [35,36]. In bioinformatics, k-mers are subsequences of length k that are included in a biological sequence. For different values of k, different sequence fragments will be obtained. The larger the value of k, the more short repeats can be crossed by k-mers. The smaller the number of k-mers, the poorer the association between k-mers. In practice, it is necessary to make reasonable selection and repeated attempts according to the genomic characteristics of the species, the size of sequencing data, and other factors.

For a peptide sequence $D : D = D_1 D_2 \cdots D_L$, where Di $\in \{A, C, \cdots, Y\}$, L is the sequence length. The principle of k-mer is to calculate the frequency of polypeptide, so there is no need to supplement the sequence, only the 20 amino acids contained in the peptide sequence itself are used. Given a sliding window size of k and a sliding step size of 1, $L - k + 1$ k-mer sequences can be obtained, and the frequency of each k-mer can be calculated as follows:

$$\varphi_i = \frac{\beta_i}{\sum_{i=1}^{20^k} \beta_i} = \frac{\beta_i}{L - k + 1} \tag{1}$$

where β_i denotes the number of occurrence of the i-th k-mer. Thus, the peptide sequence can be represented by the following k-mer formula:

$$K = [\varphi_1, \varphi_2, \cdots, \varphi_{20^k}] \tag{2}$$

In this paper, due to the computational complexity and model performance, we set k to 2, then 20^2 kinds of sequence fragments can be obtained.

A peptide sequence can then be represented as a 400-dimensional numeric vector. If you want to use k-mer to represent a peptide sequence with 22 amino acids, you only need to change the type of sequence fragment to 22^k. Then, the peptide sequence can be represented as $K = [\varphi_1, \varphi_2, \cdots, \varphi_{22^k}]$

2.3 The architecture of the classifier

AMP-EF is an ensemble framework composed of traditional machine learning and deep learning. It consists of two branches, the first branch is XGBoost, and the second branch is mainly composed of Bi-LSTM and attention mechanism. Firstly, we input the sequence features obtained above into the two branches to calculate the prediction probability, respectively. Since neural network can only receive 3-dimensional input data, we must first transform the 2-dimensional sequence features into 3-dimensional, while the input of XGBoost remains 2-dimensional. Soft voting is then used to combine the outputs of the two branches to get the final prediction, with the two branches having equal weight. The prediction accuracy of the two-branch structure is usually higher than that of the single-branch structure. The following is a detailed look at these two branches.

2.3.1 XGBoost

XGBoost is a type of boosting algorithm, the basic idea of which is to ensemble many tree models together to form a stronger classifier. Because of its high efficiency and portability, it is widely used in classification, regression, data mining, etc [37]. The objective function of XGBoost consists of a loss function and a regularization term that inhibits the complexity of the model [38], which is defined as follows:

$$F = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{i=1}^{t} \Omega(f_i) \tag{3}$$

where n denotes the number of samples, y_i denotes the true value of the i-th sample, $\hat{y}_i$ denotes the predicted value of the i-th sample, and t denotes the number of decision trees. $l(y_i, \hat{y}_i)$ is the loss function, which is represented by the predicted value and the true value of the sample. $\Omega(f_i)$ is the regularization term, which is essentially the model complexity of

the t-th tree. Summing the complexity of t trees as a regularization term is beneficial to reduce the model complexity and prevent overfitting. In general, the larger the number of trees, the better the learning ability of the model. In this study, in order to improve the prediction performance of the model as much as possible, we set n_estimator to 1000. In addition, we set max_depth to 6 and learning rate to 0.1 to avoid overfitting and use the default values for the other parameters.

2.3.2 Bi-LSTM and attention mechanism

The second branch of the classifier is deep learning model which is composed of Bi-LSTM, attention mechanism, and fully connected network. LSTM is a kind of recurrent neural network (RNN), which is suitable for processing sequence data and simulating the dependency relationship between data [39]. But compared with ordinary RNN, it introduces gate mechanism to avoid gradient vanishing and gradient explosion, and also adds memory unit to capture long distance dependence. The specific calculation formula is as follows:

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\ f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\ c_t &= f_t \otimes c_{t-1} + i_t \otimes \tilde{c}_t \\ \tilde{c}_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c) \\ h_t &= o_t \otimes c_{t-1} \end{aligned} \tag{4}$$

where σ is the sigmoid function. i, f, o, c and $\tilde{c}$ are respectively input gate, forgetting gate, output gate, unit vector, and candidate state. h is the hidden vector. Both W and U are weight matrices. $\otimes$ is the cross product. However, one-way LSTM processes sequence information strictly in chronological order, ignoring the backward dependence of sequence. In this paper, the sequence length of AMPs is 100-bp and the amino acids in the context are continuous and tight. Therefore, Bi-LSTM is selected for feature extraction. Bi-LSTM is essentially a combination of two layers of LSTMs. The forward LSTM extracts the features of the sequence forward to obtain the feature vector $h_L = [h_{L,0}, h_{L,1}, \cdots, h_{L,T}]$, and the

backward LSTM extracts the features of the sequence backward to obtain the feature vector $h_R = [h_{R,0}, h_{R,1}, \cdots, h_{R,T}]$. The results of the corresponding positions of the two are concatenated to obtain the final output $h = [h_{L,0}, h_{R,T}, h_{L,1}, h_{R,T-1}, \cdots, h_{L,T}, h_{R,0}]$. In this way, the t-th feature can be made to have both past and future information. The prediction performance of Bi-LSTM is generally better than LSTM for context-sensitive text data. Subsequently, the features generated by Bi-LSTM are fed into attention layer.

We adopt attention mechanism mentioned by Wang et al. [39]. The main idea of attention mechanism is to focus the important information with high weight and ignore the irrelevant information with low weight [40]. Inside attention layer, the dense layer with softmax activation function is first used to calculate the weight of each item in the input sequence, and then the weight and the input bitwise correspondence are multiplied to obtain a new feature vector. The important sequence segments of AMPs are similar to some extent, and the use of attention layer can focus more attention on the consistent seed sequence, thus improving the prediction performance.

In this part, Bi-LSTM layer and attention layer are mainly used for feature selection, to make the features obtained by one-hot encoding and 2-mer cleverly fused together. Bi-LSTM layer can deepen the connection between context features, and attention layer can extract key features. Finally, a 3-layer fully connected layer is used to decode the features and complete the final identification. We set the number of recurrent units in Bi-LSTM to 64 (32 forward and 32 backward), attention layer has the same input and output dimensions, and a dropout layer is added between the dense layer with 64 neurons and 32 neurons to prevent overfitting. This is then passed to another dense layer with sigmoid function to get the probability of the sample being positive and negative respectively. The loss function is categorical_crossentropy loss. The detailed parameters of this branch are shown in Table 2.

Table 2. The parameters of the second branch of the classifier.

Layers	Parameters
Bi-LSTM	hidden_size=64, dropout ratio=0.3
Attention	Default
Dense 1	units=64, kernel_initializer='RandomNormal', activation='relu'
Dropout	dropout ratio=0.4
Dense 2	units=32, kernel_initializer='RandomNormal', activation='relu'
Dense 3	units=2, activation='sigmoid'

2.4 Model evaluation

In this study, five-fold cross validation and independent dataset testing are used to test the effectiveness and generalization of the model. Our model finally obtains the probability values of the sample to be positive and negative respectively, and then compares the two, and selects the category corresponding to the larger probability value as the final prediction result of the sample. In order to fairly evaluate our model, we use the following five evaluation metrics as in previous studies: accuracy (ACC), sensitivity (Sn), specificity (Sp), Matthew's correlation coefficient (MCC), and area under ROC curve (AUC) [41–43]. Among them, we use ACC as the main metric to train and evaluate the model. The formulas are as follows:

$$\begin{aligned} ACC &= \frac{S_p^+ + S_n^+}{S_p^+ + S_n^+ + S_p^- + S_n^-} \\ Sn &= \frac{S_p^+}{S_p^+ + S_n^-} \\ Sp &= \frac{S_n^+}{S_p^- + S_n^+} \\ MCC &= \frac{S_p^+ \times S_n^+ - S_p^- \times S_n^-}{\sqrt{(S_p^+ + S_n^-)(S_p^+ + S_p^-)(S_n^+ + S_p^-)(S_n^+ + S_n^-)}} \end{aligned} \tag{5}$$

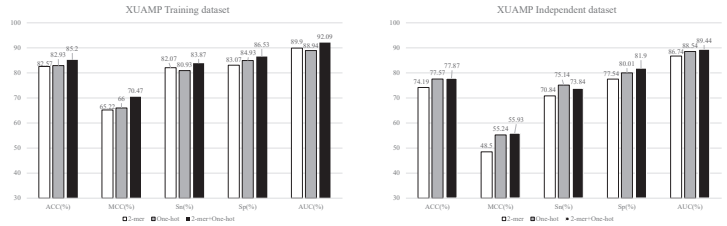
where S_p^+, S_n^+, S_p^-, and S_n^- represent correctly predicted positive samples, negative samples, and incorrectly predicted positive samples, negative samples, respectively. ACC is the most simple and intuitive evaluation index in identification problems, which indicates the proportion of correctly classified samples in the total number of samples. Sn and Sp represent the probabilities of correctly predicting positive and negative samples, respectively. MCC is essentially a correlation coefficient describing the actual

result and the predicted result, and it works equally well on imbalanced datasets. AUC is the area under the ROC curve, between 0 and 1, to intuitively evaluate the quality of the classifier. The bigger, the better.

3 Results and discussion

3.1 Comparison of feature extraction methods

In this study, we use both one-hot coding and k-mer to extract features. They can get different peptide sequence information, respectively. One-hot coding is based on the position information of peptide sequence, while k-mer is based on the composition and position of amino acids. To illustrate that both feature extraction methods are helpful for representing AMPs sequences, we compare the prediction performance of single features and combined features on the XUAMP training dataset and independent test dataset. As shown in Figure 2, on the training dataset, the ACC value of combined features reaches 85.2%, which is 2.27% and 2.63% higher than that of one-hot coding and k-mer, and other evaluation indicators are also improved compared with single features. On the independent test dataset, the ACC value of combined features is 3.68% higher than that of k-mer, but there is little difference between the result of one-hot encoding. To this end, experiments are conducted on LAMP [28, 29], dbAMP [31], DRAMP [24, 25], and DBAASP [16] independent test datasets. The results show that the ACC values of combined features are 2.38%, 0.38%, 1.21%, and 0.85% higher than that of single one-hot encoding, respectively. The MCC values are increased by 4.57%, 0.75%, 2.61%, and 1.26%, respectively. It can be seen that the addition of k-mer can improve the generalization of the model so that it performs well on other datasets. The specific results are described in Figure 3.



(a) XUAMP training dataset (b) XUAMP independent test dataset

Figure 2. Comparison of different feature extraction methods on the XUAMP training dataset and independent test dataset.

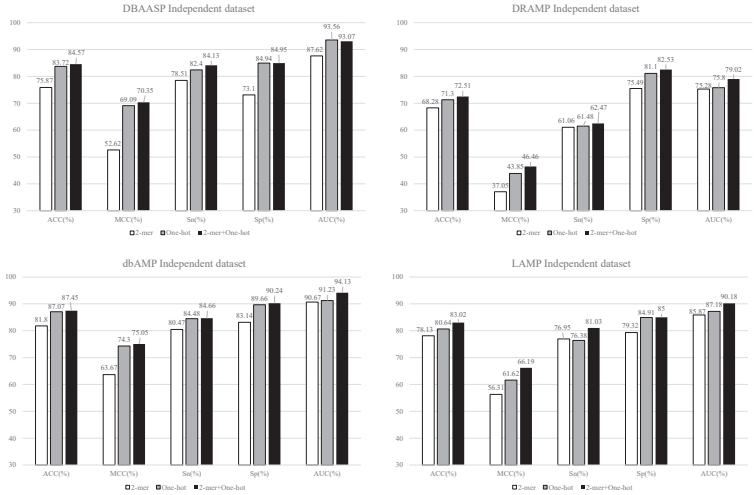
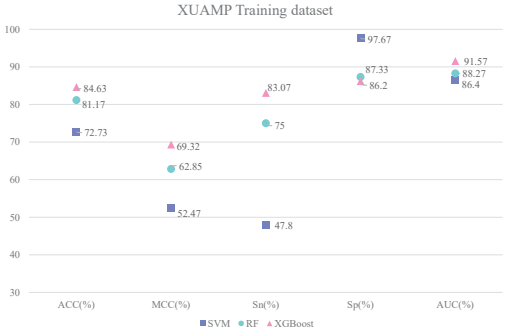


Figure 3. Comparison of different feature extraction methods on other independent test datasets.

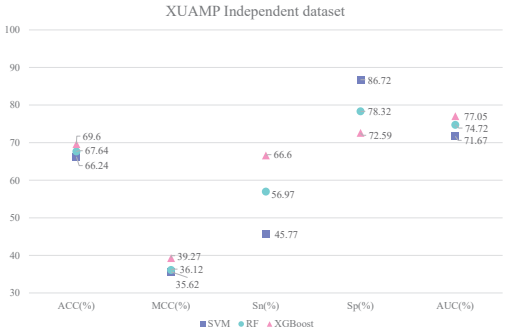
3.2 Analysis of the two-branch ensemble framework

For the first branch of the classifier, we adopt XGBoost. It is a commonly used machine learning algorithm in identification problems and is capable of handling high-dimensional data features. Previously, most researchers used SVM and RF to predict AMPs. We compare these three machine learning methods on the XUAMP training dataset and independent test dataset, and the results are shown in Figure 4. (SVM parameters: probability is 'True', kernel is 'poly'; RF parameters: n_estimators is

500, criterion is 'gini', max_depth is 10.) The Sp value of SVM is indeed significantly higher than the other two methods, but the other indicators are lower than the other two methods. On the training dataset, the ACC, MCC, Sn, and AUC values of XGBoost are 84.63%, 69.32%, 83.07%, and 91.57%, respectively, which achieve the highest values, and the Sp value is slightly lower than RF. On the independent test dataset, the ACC, MCC, and AUC values of XGBoost are 69.6%, 39.27%, and 77.05%, respectively, which are slightly better than other methods. And the Sn value of XGBoost is significantly improved, reaching 66.6%. Overall, XGBoost has the edge.



(a) XUAMP training dataset



(b) XUAMP independent test dataset

Figure 4. Comparison of different machine learning methods on the XUAMP training dataset and independent test dataset.

We use Bi-LSTM with attention mechanism as the second branch of

the classifier. Bi-LSTM layer takes the fusion feature vector as the input and extracts advanced features from each residue in both forward and reverse directions respectively, which can fully mine the correlation information between residues. By using attention layer to increase the weight of important information, the prediction accuracy can be further improved. In this part, we compare LSTM, Bi-LSTM, LSTM+Attention, Bi-LSTM+Attention, and AMP-EF, and the results are shown in Figure 5. On the XUAMP training dataset, the ACC values of LSTM, Bi-LSTM, LSTM+Attention, and Bi-LSTM+Attention are 76.67%, 80.67%, 79.43%, and 82.20%, respectively. Bi-LSTM is 4% higher than LSTM, LSTM+Attention is 2.76% higher than LSTM, and Bi-LSTM+Attention is 5.53% higher than LSTM. It can be seen that the addition of Bi-LSTM layer and attention layer is indeed more beneficial to identify AMPs. On the XUAMP independent test dataset, the ACC values of the four models are 65.14%, 66.86%, 66.64%, and 67.97%, respectively. The evaluation indexes of Bi-LSTM+Attention are the highest among the four models.

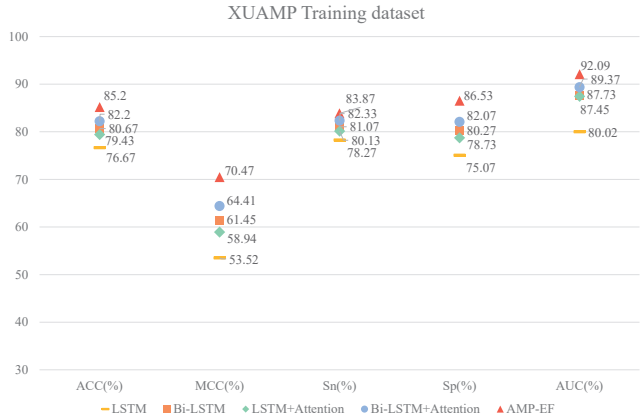
In addition, to more clearly illustrate the superiority of the two-branch ensemble structure, we present the results of XGBoost, Bi-LSTM+Attention, and AMP-EF in Table 3.

Table 3. Comparison of two-branch and single-branch structures on the XUAMP training dataset and independent test dataset.

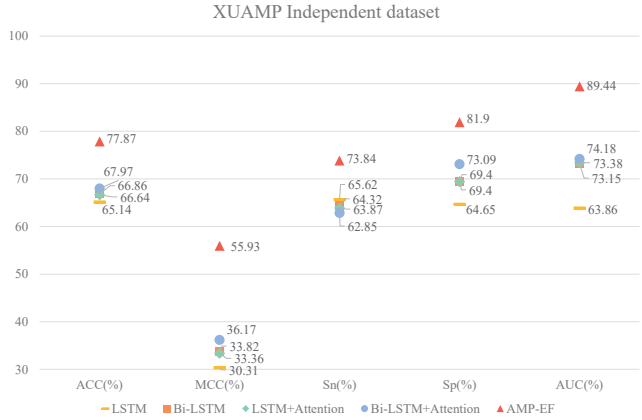
Dataset	Model	ACC(%)	MCC(%)	Sn(%)	Sp(%)	AUC(%)
Training	XGBoost	84.63	69.32	83.07	86.20	91.57
	Bi-LSTM+Attention	82.20	64.41	82.33	82.07	89.37
	AMP-EF	85.20	70.47	83.87	86.53	92.09
Independent	XGBoost	69.60	39.27	66.60	72.59	77.05
	Bi-LSTM+Attention	67.97	36.17	62.85	73.09	74.18
	AMP-EF	77.87	55.93	73.84	81.90	89.44

3.3 Comparison with existing methods

In the previous identification of AMPs, researchers usually only used a single machine learning method or a single deep learning method, and its effect still has a lot of room for improvement. AMP-EF is a two-branch ensemble model composed of XGBoost, Bi-LSTM, and attention mechanism. Its advantage is that it uses machine learning and deep learning



(a) XUAMP training dataset



(b) XUAMP independent test dataset

Figure 5. Comparison of different combinations of deep learning methods on the XUAMP training dataset and independent test dataset.

at the same time, and the two complement each other to further improve the prediction performance. Usually, deep learning algorithms need to continuously learn with a large amount of data, while traditional machine learning algorithms use formulated rules and are able to adapt to various amounts of data, especially small amounts. On the other hand, the performance of most machine learning algorithms depends on the accuracy

of the extracted features, but deep learning can directly obtain high-level features from data, reducing the requirements for feature representation methods. Effectively fusing the two can not only improve the prediction accuracy of AMPs but also enhance the generalization of the model, so that it also performs well on different datasets. To this end, we compare the performance of AMP-EF with other existing methods on the XUAMP independent test dataset, and the results are shown in Table 4. The ACC, MCC, Sn, and AUC values of AMP-EF are 77.9%, 55.9%, 73.8%, and 89.4%, respectively, which are 6.4%, 9.5%, 20.8%, and 11.7% higher than those of the best existing models. The Sp value, while not reaching the optimum, is also highly comparable. In addition, we also compare ACC, MCC, and AUC values on the other seven independent test datasets. Our model outperforms the best existing models on CAMP [26,27], APD3 [30], and YADAMP [23] datasets and achieves highly comparable results on LAMP [28,29], dbAMP [31], DRAMP [24,25], and DBAASP [16] datasets. Detailed results are shown in Table 5 to 7. The results for other existing models are from Yan et al. [16].

Table 4. Comparison with existing methods on the XUAMP independent test dataset.

Model	ACC(%)	MCC(%)	Sn(%)	Sp(%)	AUC(%)
amPEPpy	67.9	43.1	40.0	95.8	74.2
AMPfun	67.4	41.4	40.6	94.3	73.5
AMPEP	66.1	42.9	33.0	99.2	72.7
ADAM-HMM	68.4	39.0	52.1	84.7	68.4
ampir	56.3	15.6	26.6	85.9	61.9
AMPScannerV2	56.8	13.7	52.3	61.3	58.5
AmpGram	56.4	13.1	44.5	68.2	54.7
Deep-AMPEP30	53.3	18.3	6.5	100	53.3
CAMP-ANN	58.4	18.2	38.5	78.2	58.4
sAMPpred-GAT	71.5	46.4	53.0	90.0	77.7
AMP-EF	77.9	55.9	73.8	81.9	89.4

Table 5. ACC values for existing methods and AMP-EF on the other seven independent test datasets.

Model	YADAMP	DRAMP	CAMP	LAMP	APD3	dbAMP	DBAASP
amPEPpy	91.5	73.4	94.8	76.5	93.9	88.9	–
AMPfun	95.4	73.9	96.8	75.9	91.6	86.3	–
AMPEP	96.9	71.2	97.3	75.5	93.6	76.6	–
ADAM-HMM	92.7	73.6	86.9	87.2	88.6	88.6	–
ampir	56.6	57.7	50.0	61.4	60.7	62.4	–
AMPScannerV2	76.1	64.6	71.7	67.7	79.9	76.9	–
AmpGram	77.8	62.0	80.5	68.3	76.8	76.6	–
Deep-AMPEP30	73.1	53.0	100	57.8	60.9	59.3	–
CAMP-ANN	73.3	63.8	67.5	64.8	74.5	71.6	–
sAMPpred-GAT	95.5	76.0	95.6	84.0	89.6	88.8	–
AMP-EF	98.5	72.5	99.8	83.0	92.4	87.5	84.6

Table 6. MCC values for existing methods and AMP-EF on the other seven independent test datasets.

Model	YADAMP	DRAMP	CAMP	LAMP	APD3	dbAMP	DBAASP
amPEPpy	83.0	50.6	89.7	55.7	87.9	78.0	–
AMPfun	91.1	51.1	93.8	54.6	83.2	73.0	–
AMPEP	93.8	49.5	94.6	55.8	87.3	77.9	–
ADAM-HMM	85.6	49.5	74.2	74.4	77.1	77.2	–
ampir	17.1	19.7	0.0	27.9	27.2	29.0	–
AMPScannerV2	53.5	29.2	43.7	35.4	62.2	55.1	–
AmpGram	56.6	24.3	62.6	36.5	54.8	54.4	–
Deep-AMPEP30	54.9	17.5	100	29.1	35.0	31.7	–
CAMP-ANN	46.7	28.7	36.1	30.3	49.2	43.6	–
sAMPpred-GAT	91.2	55.7	91.6	69.4	79.3	78.0	–
AMP-EF	96.9	46.5	99.5	66.2	85.0	75.1	70.3

Table 7. AUC values for existing methods and AMP-EF on the other seven independent test datasets.

Model	YADAMP	DRAMP	CAMP	LAMP	APD3	dbAMP	DBAASP
amPEPpy	96.8	75.9	97.8	85.5	97.2	94.0	87.0
AMPfun	99.7	81.0	100	85.2	97.2	93.0	93.0
AMPEP	99.2	77.3	99.4	81.8	98.3	93.3	92.2
ADAM-HMM	92.7	73.6	86.9	87.2	88.6	88.6	–
ampir	74.5	63.0	75.1	73.7	75.8	75.2	–
AMPScannerV2	84.1	67.9	77.3	72.9	88.4	85.1	–
AmpGram	87.5	64.0	91.6	71.3	86.0	86.6	–
Deep-AMPEP30	73.1	53.0	100	57.8	60.9	59.3	–
CAMP-ANN	73.3	63.8	67.5	64.8	74.5	71.6	–
sAMPpred-GAT	99.3	82.8	100	92.5	95.5	95.4	94.2
AMP-EF	99.7	79.0	100	90.2	95.7	94.1	93.1

4 Conclusion

AMPs are expected to be developed as new antibiotics to treat human and animal diseases to address the problem of microbial resistance to antibiotics. Therefore, accurate identification of AMPs is helpful to the development of the pharmaceutical industry. Previous identification models of AMPs are relatively simple. In this paper, we propose AMP-EF, a two-branch ensemble framework based on machine learning and deep learning. In the feature representation part, we use one-hot encoding based on peptide sequence position information and k-mer based on amino acid composition. Then the two-branch structure consisting of XGBoost and Bi-LSTM+Attention is used as the classifier. In this way, machine learning and deep learning methods can complement each other and bring out the best in each other. On the XUAMP training dataset and independent test dataset, the ACC values of our model are 85.20% and 77.87%, respectively. The ACC values of the other seven independent test datasets are 98.5%, 72.5%, 99.8%, 83.0%, 92.4%, 87.5%, and 84.6%, respectively. Hopefully, our model can help biologists and medical professionals identify and study AMPs more effectively.

***Acknowledgment*:** This work was supported by the National Natural Science Foundation of China (No.12101480), the Fundamental Research Funds for the Central Universities (No.QTZX23002), and the Opening Foundation of Key Laboratory of Computational Science and Application of Hainan Province (No.JSKX 202301).

References

[1] S. Panwar, M. Thapliyal, V. Kuriyal, V. Tripathi, A. Thapliyal, GEU-AMP50: Enhanced antimicrobial peptide prediction using a machine learning approach, *Mater. Today Proceed.* **73** (2023) 81–87.

[2] H. Wang, M. C. Niu, T. Xue, L. H. Ma, X. L. Gu, G. C. Wei, F. Q. Li, C. H. Wang, Development of antibacterial peptides with efficient antibacterial activity, low toxicity, high membrane disruptive activity and a synergistic antibacterial effect, *J. Mater. Chem. B* **10** (2022) 1858–1874.

[3] W. Y. Zhou, Y. F. Liu, Y. X. Li, S. Q. Kong, W. L. Wang, B. Y. Ding, J. Y. Han, C. Z. Mou, X. Gao, J. T. Liu, TriNet: A tri-fusion neural network for the prediction of anticancer and antimicrobial peptides, *Patterns* **4** (2023) #100702.

[4] D. F. Christopher, E. W. H. Robert, A. Cherkasov, AMPer: a database and an automated discovery tool for antimicrobial peptides, *Bioinformatics.* **23** (2007) 1148–1155.

[5] S. Joseph, S. Karnik, P. Nilawe, V. K. Jayaraman, S. Idicula-Thomas, ClassAMP: a prediction tool for classification of antimicrobial peptides, *IEEE ACM T. Comput. Bi.* **9** (2012) 1535–1538.

[6] T. J. Lawrence, D. L. Carper, M. K. Spangler, A. A. Carrell, T. A. Rush, S. J. Minter, D. J. Weston, J. L. Labbé, amPEPpy 1.0: A portable and accurate antimicrobial peptide prediction tool, *Bioinformatics* **37** (2020) 2058–2060.

[7] P. Bhadra, J. L. Yan, J. Y. Li, S. Fong, W. I. S. Shirley, AmPEP: sequence-based prediction of antimicrobial peptides using distribution patterns of amino acid properties and random forest, *Sci. Rep.* **8** (2018) #1697.

[8] C. Chia-Ru, K. Ting-Rung, W. Li-Ching, L. Tzong-Yi, H. Jorng-Tzong, Characterization and identification of antimicrobial peptides with different functional activities, *Brief. Bioinf.* **21** (2020) 1098–1114.

[9] B. Michal, S. Katarzyna, R. Dominik, P. Filip, C. Jaroslaw, R. Stefan, G. Przemyslaw, Proteomic screening for prediction and design of antimicrobial peptides with AmpGram, *Int. J. Mol. Sci.* **21** (2020) #4310.

[10] L. C. H. W. Fingerhut, D. J. Miller, J. M. Strugnell, N. L. Daly, I. R. Cooke, ampir: an R package for fast genome-wide prediction of antimicrobial peptides, *Bioinformatics* **36** (2020) 5262–5263.

[11] K. Kavousi, M. Bagheri, S. Behrouzi, S. Vafadar, A. F. Fallah, L. B. Teimoori, S. Ariaeenejad, A. Shockravi, A. A. Moosavi Movahedi, IAMPE: NMR assisted computational prediction of antimicrobial peptides, *J. Chem. Inf. Model.* **60** (2020) 4691–4701.

[12] D. Veltri, U. Kamath, A. Shehu, Deep learning improves antimicrobial peptide recognition, *Bioinformatics* **34** (2018) 2740–2747.

[13] X. Su, J. Xu, Y. B. Yin, X. W. Quan, H. Zhang, Antimicrobial peptide identification using multi-scale convolutional network, *BMC Bioinf.* **20** (2019) #730.

[14] J. L. Yan, P. Bhadra, A. Li, P. Sethiya, L. G. Qin, H. K. Tai, K. H. Wong, S. W. I. Siu, Deep-AmPEP30: Improve short antimicrobial peptides prediction with deep learning, *Mol. Therapy Nucleic Acids.* **20** (2020) 882–894.

[15] C. K. Li, D. Sutherland, S. A. Hammond, C. Yang, F. Taho, L. Bergman, S. Houston, R. L. Warren, T. Wong, L. M. N. Hoang, C. E. Cameron, C. C. Helbing, I. Birol, AMPlify: attentive deep learning model for discovery of novel antimicrobial peptides effective against WHO priority pathogens, *BMC Genomics* **23** (2022) #77.

[16] K. Yan, H. W. Lv, Y. C. Guo, W. Peng, B. Liu, sAMPpred-GAT: Prediction of antimicrobial peptide by graph attention network and predicted peptide structure, *Bioinformatics.* **39** (2022) #btac715.

[17] R. GarciaJacas Cesar, S. A. PinachoCastellanos, L. A. GarciaGonzalez, C. A. Brizuela, Do deep learning models make a difference in the identification of antimicrobial peptides, *Brief. Bioinf.* **23** (2022) 1–16.

[18] X. Y. Wang, F. Y. Li, J. Xu, J. Rong, G. I. Webb, Z. Y. Ge, J. Li, J. N. Song, ASPIRER: a new computational approach for identifying non-classical secreted proteins based on deep learning, *Brief. Bioinf.* **23** (2022) 1–12.

[19] J. Xu, F. Y. Li, A. Leier, D. X. Xiang, H. H. Shen, L. T. T. Marquez, J. Li, D.J. Yu, J. N. Song, Comprehensive assessment of machine learning-based methods for predicting antimicrobial peptides, *Brief. Bioinf.* **22** (2021) 1–22.

[20] Y. Michael, S. Christian, Q. Peng, Long short-term memory recurrent neural networks for antibacterial peptide identification, *IEEE Xplore* **5** (2017) 498–502.

[21] C. Wang, S. Garlick, M. Zloh, Deep learning for novel antimicrobial peptide design, *Biomolecules* **11** (2021) #471.

[22] R. Sharma, S. Shrivastava, S. S. Kumar, A. Kumar, S. Saxena, S. R. Kumar, Deep-ABPpred: identifying antibacterial peptides in protein sequences using bidirectional LSTM with word2vec, *Brief. Bioinf.* **22** (2021) 1–19.

[23] P. P. Stefano, S. Lucia, C. Simona, I. Pio, YADAMP: yet another database of antimicrobial peptides, *Int. J. Antimicrob. Agents* **39** (2012) 346–351.

[24] L. L. Fan, J. Sun, M. F. Zhou, J. Zhou, X. Z. Lao, H. Zheng, H. M. Xu, DRAMP: a comprehensive data repository of antimicrobial peptides, *Sci. Rep.* **6** (2016) #24482.

[25] X. Y. Kang, F. Y. Dong, C. Shi, S. C. Liu, J. Sun, J. X. Chen, H. Q. Li, H. M. Xu, X. Z. Lao, H. Zheng, DRAMP 2.0, an updated data repository of antimicrobial peptides, *Sci. Data.* **6** (2019) #148.

[26] S. N. Thomas, S. Karnik, R. S. Barai, V. K. Jayaraman, S. Idicula-Thomas, CAMP: a useful resource for research on antimicrobial peptides, *Nucleic Acids Res.* **38** (2010) D774–D780.

[27] F. H. Waghu, R. S. Barai, P. Gurung, S. Idicula-Thomas, CAMPR3: a database on sequences, structures and signatures of antimicrobial peptides, *Nucleic Acids Res.* **44** (2016) D1094–1097.

[28] G. Z. Ye, H. Y. Wu, J. J. Huang, W. Wang, K. K. Ge, G. D. Li, J. Zhong, Q. S. Huang, LAMP2: a major update of the database linking antimicrobial peptides, *Database (Oxford)* (2020) #baaa061.

[29] X. W. Zhao, H. Y. Wu, H. R. Lu, G. D. Li, Q. S. Huang, LAMP: A database linking antimicrobial peptides, *PLoS One* **8** (2013) #e66557.

[30] G. Wang, X. Li, Z. Wang, APD3: the antimicrobial peptide database as a tool for research and education, *Nucleic Acids Res.* **44** (2016) D1087–1093.

[31] J. H. Jhong, Y. H. Chi, W. C. Li, T. H. Lin, K. Y. Huang, T. Y. Lee, dbAMP: an integrated resource for exploring antimicrobial peptides with functional activities and physicochemical properties on transcriptome and proteome data, *Nucleic Acids Res.* **47** (2019) D285–D297.

[32] W. Li, A. Godzik, Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences, *Bioinformatics* **22** (2006) 1658–1659.

[33] M. Novič, M. Randić, Representation of proteins as walks in 20-D space, *SAR QSAR Environ. Res.* **19** (2008) 317–337.

[34] A. Czerniecka, D. Bieliñska-Wąz, P. Wąz, T. Clark, 20D-dynamic representation of protein sequences, *Genomics* **107** (2016) 16–23.

[35] 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 T. Comput. Bi.* **20** (2023) 1926–1934.

[36] B. Manavalan, S. Basith, T. H. Shin, D. Y. Lee, L. Y. Wei, G. Lee, 4mCpred-EL: An ensemble learning framework for identification of DNA N4-methylcytosine sites in the mouse genome, *IEEE ACM T. Comput. Bi.* **8** (2019) #1332.

[37] T. Q. Chen, C. Guestrin, XGBoost: A scalable tree boosting system, *CoRR.* **abs/1603.02754** (2016) 785–794.

[38] B. S. Ma, G. Yan, B. J. Chai, X. Y. Hou, XGBLC: An Improved Survival Prediction Model Based on XGBoost, *Bioinformatics* **38** (2021) 410–418.

[39] L. Y. Wang, S. G. Zheng, H. Zhang, Z. Y. Qiu, X. D. Zhong, H. M. Liu, Y. N. Liu, ncRFP: A novel end-to-end method for non-coding RNAs family prediction based on Deep Learning, *IEEE ACM T. Comput. Bi.* **18** (2020) 784–789.

[40] Z. W. Ma, J. P. Zhao, J. Tian, C. H. Zheng, DeeProPre: A promoter predictor based on deep learning, *Comput. Biol. Chem.* **101** (2022) #107770.

[41] S. Basith, B. Manavalan, T. H. Shin, G. Lee, iGHBP: Computational identification of growth hormone binding proteins from sequences using extremely randomised tree, *Comput. Struct. Biotech. J.* **16** (2018) 412–420.

[42] B. Manavalan, S. Basith, T. H. Shin, L. Y. Wei, G. Lee, mAHTPred: a sequencebased meta-predictor for improving the prediction of anti-hypertensive peptides using effective feature representation, *Bioinformatics* **35** (2019) 2757–2765.

[43] H. J. Qiao, S. L. Zhang, T. Xue, J. S. Wang, B. W. Wang, iPro-GAN: A novel model based on generative adversarial learning for identifying promoters and their strength, *Comput. Meth. Prog. Bio.* **215** (2022) #106625.