Identifying Phage Sequences from Metagenomic Data Using Deep Neural Network with Word Embedding and Attention Mechanism

Lijia Ma, Wenwei Deng, Yuan Bai, Zhanwei Du, Minfeng Xiao, Lin Wang, *Senior Member, IEEE,* Jianqiang Li, *Member, IEEE,* and Asoke K. Nandi, *Fellow, IEEE*

***Abstract*—Phages are the functional viruses that infect bacteria and they play important roles in microbial communities and ecosystems. Phage research has attracted great attention due to the wide applications of phage therapy in treating bacterial infection in recent years. Metagenomics sequencing technique can sequence microbial communities directly from an environmental sample. Identifying phage sequences from metagenomic data is a vital step in the downstream of phage analysis. However, the existing methods for phage identification suffer from some limitations in the utilization of the phage feature for prediction, and therefore their prediction performance still need to be improved further. In this article, we propose a novel deep neural network (called MetaPhaPred) for identifying phages from metagenomic data. In MetaPhaPred, we first use a word embedding technique to encode the metagenomic sequences into word vectors, extracting the latent feature vectors of DNA words. Then, we design a deep neural network with a convolutional neural network (CNN) to capture the feature maps in sequences, and with a bi-directional long short-term memory network (Bi-LSTM) to capture the long-term dependencies between features from both forward and backward directions. The feature map consists of a set of feature patterns, each of which is the weighted feature extracted by a convolution filter with convolution kernels in the CNN slide along the input feature vectors. Next, an attention mechanism is used to enhance contributions of important features. Experimental results on both simulated and real metagenomic data with different lengths demonstrate the superiority of the proposed MetaPhaPred over the state-of-the-art methods in identifying phage sequences. The source code and data are available at: https://github.com/dww01/MetaPhaPred.**

***Index Terms*—deep learning, metagenomics, phage identification, word embedding, attention mechanism.**

I. Introduction

PHAGES, viruses that infect bacteria and archaea, are considered as the most abundant and diverse biological entities on the planet with an estimated 10^{31} phage particles [1], [2]. Phages play important roles in microbial communities and ecosystems, such as human gut and the ocean [1], [3]. By infecting bacteria in human body, phages can directly affect human health and disease [4], [5]. Phage therapy uses phages as therapeutic agents to treat bacterial infections, and has attracted significant research attention in recent years [6]–[8]. In the past decades, phages have been discovered by the culture-dependent genome sequencing methods, which are time consuming and expensive. The metagenomics sequencing technique has recently emerged as a novel genome sequencing method, which is capable of sequencing microbial communities directly from an environmental sample. However, the sequences obtained by the metagenomics sequencing technique generally contain a mixture of phage and bacteria genomic sequences [9]. Identifying the phage sequences from metagenomic data can effectively help phage genome assembly and increase the genome completeness, and is a vital step in the downstream of phage analysis and novel phage discovery [10]–[12]. However, metagenomic phage sequence identification remains a challenging problem.

Certain methods for phage identification in metagenomic data have been proposed. These methods can be classified into three categories: sequence alignment-based, machine learning-based, and deep learning-based methods. The sequence alignment-based methods rely on sequence alignment against the previously built phage databases. For examples, VirSorter [13] built viral reference databases and compared the sequences with more than two detected genes to the database to identify viral contigs. MetaPhinder [14] built a database of phage whole genome sequences and compared the contigs to the database using BLAST (Basic Local Alignment Search Tool) [15] to identify phage contigs. However, these sequence alignment-based methods find it challenging to identify short phage sequences.

The machine learning-based methods extract the pre-defined sequence features and train a classifier for phage identification, achieving better performances than the sequence alignment-based methods. For examples, VirFinder [12] computed the k-mer frequency of the contigs and trained a logistic regression model to identify viral sequences. VIBRANT [16] extracted protein features and used a hybrid machine learning model for virus identification. VirSorter2 [17] was an extension of VirSorter and trained random forest classifiers based on the gene features to identify viruses.

This work was supported by the National Key R&D Program of China under Grant 2020YFA0908700, by the National Natural Science Foundation of China under Grants 62173236, 62176164, 62203134, 62073225, 62203134, 61972263, and 62072315, and by the Technology Research Project of Shenzhen City under Grant JCYJ20190808174801673.

L. Ma, W. Deng, and J. Li are with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China.

Y. Bai and Z. Du are with the University of Hong Kong, Hong Kong, China.

M. Xiao is with BGI-Shenzhen, Shenzhen 518083, China.

L. Wang is with University of Cambridge, Cambridge CB2 3EH, United Kingdom.

A. K. Nandi is with the Department of Electronic and Electrical Engineering, Brunel University London, Uxbridge, UB8 3PH, United Kingdom and also a Distinguished Visiting Professor at Shenzhen University, China.

(Corresponding authors: Yuan Bai (email: ybai0523@outlook.com) and Jianqiang Li (e-mail: lijq@szu.edu.cn).)

As deep learning has achieved great success in many fields in recent years, certain deep learning-based methods have been proposed for metagenomic phage identification, which can automatically extract and learn features for classification. For examples, PPR-Meta [18] designed a bi-path convolutional neural network (CNN) to identify phage and plasmid sequences from the metagenomic data. DeepVirFinder [19] used a CNN to predict prokaryotic viral sequences. VirNet [20], Seeker [21], and RNN-VirSeeker [22] designed a long short-term memory network (LSTM) for phage identification. Virtifier [23] employed an attention-based LSTM for short viral sequences identification. PhaMer [24] converted DNA sequences into protein-token sentences and designed a transformer-based model for phage identification.

However, the existing methods still have some limitations and their prediction performances need to be improved further. The machine learning-based methods and some of the deep learning-based methods required features extraction from the raw sequences artificially, and how to appropriately design and select the features remains a challenging problem. Moreover, the deep learning-based methods mainly used the one-hot encoding technique to represent DNA sequences, neglecting the correlation information between nucleotides. In addition, the single CNN or LSTM architecture may fail to extract sufficient features from the sequences for a robust prediction.

In the deep neural networks, CNN is capable of extracting the feature maps, namely a set of feature patterns, by using convolution operation [25], while LSTM can effectively capture the long-term dependencies between features by introducing a gate mechanism [26]. In the CNN, a feature pattern is the weighted feature extracted by a convolution filter with convolution kernels in the CNN slide along the input feature vectors. Moreover, the word embedding is a representative feature learning technique used in natural language processing that maps words to continuous vectors of real values and realizes the distributed feature representation of words [27], [28]. In addition, the attention mechanism enables the deep neural networks to focus on the important features for prediction tasks, achieving great success in many fields, such as computer vision [29], document classification [30], and machine translation [31].

In this article, inspired by the characteristics of CNN, LSTM, word embedding, and attention mechanism, we propose MetaPhaPred (Metagenomic Phage sequences Predictor), a novel deep neural network combined with word embedding and attention mechanism, for identifying phages from metagenomic data, taking the potential feature vectors, the feature maps, and the long-term dependence between the features into consideration. In MetaPhaPred, we first use the word embedding technique dna2vec [28] to learn the distributed feature representation of k-mer words, and convert DNA sequences into word vectors. Then, we design a deep neural network with a CNN layer to capture the feature maps in sequences, and adopt a bi-directional LSTM (Bi-LSTM) layer to capture the long-term dependencies between the features in the feature maps from both forward and backward directions. Subsequently, the generated feature vector is fed into an attention layer to enhance contributions of important features. Finally, a sigmoid unit is used to compute a score for the phage prediction. Experimental results on both the simulated and real-world metagenomic sequences indicate that the proposed MetaPhaPred outperforms the state-of-the-art methods for identifying phages with different lengths.

The rest of the paper is organized as follows. Section II describes the problem definition, datasets construction, framework of the proposed MetaPhaPred, and the evaluation metrics. Section III presents the experimental results and demonstrates the effectiveness of our proposed model compared with the baseline methods. Section IV offers a brief conclusion and indicates some possible future works.

II. Materials and methods

A. Problem Definition

We let $x_i = (a_1, a_2, \ldots, a_m)$ represent the ith metagenomic sequence, where each element a_j, $1 \leq j \leq m$, denotes a nucleotide base coming from the four nucleotide bases {A, G, C, T} and m is the length of the sequence. Moreover, we let $y_i \in \{0, 1\}$ denote the classification label of x_i. More specifically, $y_i = 0$ and $y_i = 1$ denote that x_i is the bacteria/archaea sequence and the phage sequence, respectively.

Identifying phage sequences from metagenomic data can be modeled as the following binary classification task: given a number n of the input metagenomic sequences $\{x_i, y_i\}_{i=1}^{n}$, the phage identification problem tries to find a classification model to accurately predict whether the sequence x_i in metagenomic data, $1 \leq i \leq n$, is a phage sequence (i.e., $y_i = 1$) or a bacteria/archaea sequence (i.e., $y_i = 0$).

B. Datasets

We used simulated metagenomic sequences generated from genomes to construct the datasets for training and testing. We collected the genomes of 12,646 phages and 16,665 prokaryotes (bacteria and archaea) released before October 31, 2022 from the National Center for Biotechnology Information (NCBI) (https://www.ncbi.nlm.nih.gov/genome/browse\#!/overview/). The phage and bacteria/archaea genome information is provided at https://github.com/dww01/MetaPhaPred/blob/master/NCBI_genome.xlsx. To evaluate the methods' ability to identify new phages, genomes published before January 1, 2020 were used in the training sets, and the rest of genomes were used in the test sets. As the prokaryote genomes may contain prophages (i.e., integrated phages in the prokaryote genome), PhiSpy [32] was used to predict and remove prophages from all prokaryote genomes, and the prophages were incorporated into the phage training set. We randomly extracted segments from the genomes to generate simulated metagenomic contigs. Four groups of contigs with different length ranges were generated to simulate the sequences with different lengths in metagenomic data. The length ranges of contigs in each group are 100-400 bp (base pairs), 401-800 bp, 801-1200 bp, and 5000-10000 bp, simulating the sequences obtained with different sequencing technologies and long contigs, respectively. Each group has the same number of phage and prokaryote contigs. The groups with a length range of 100-400 bp, 401-800 bp and 801-1200 bp were used to train

TABLE I
THE NUMBER OF SEQUENCES IN THE SIMULATED TRAINING AND TEST DATASETS.

Sequence length	Training set		Test set	
	Phage	Bacteria	Phage	Bacteria
100-400 bp	300,000	300,000	30,000	30,000
401-800 bp	300,000	300,000	30,000	30,000
801-1200 bp	300,000	300,000	30,000	30,000
5000-10000 bp			10,000	10,000

TABLE II
THE NUMBER OF SEQUENCES IN THE PROPHAGE AND REAL METAGENOMIC PHAGE DATASET.

Datasets	Number of sequences
Prophage 100-400 bp	30,000
Prophage 401-800 bp	30,000
Prophage 801-1200 bp	30,000
Prophage 5000-10000 bp	10,000
Real metagenomic phage	107,529

three different models, respectively. The group with a length range of 5000-10000 bp was used to evaluate the performance on identifying long phage sequences. Table I summarizes the number of contigs in different groups.

We also evaluated MetaPhaPred on a prophage dataset, which was used in PPR-meta [18], containing 267 manually annotated prophages from Casjens [33] and these prophages were removed from the training set. Moreover, we used real metagenomic data to evaluate further ability of MetaPhaPred. Real bovine rumen metagenomic data, in which phages were enriched before sequencing [34], were used for evaluation. We downloaded the assembly sequences of rumen phages from PPR-meta [18]. The two datasets are summarized in Table II.

C. The Proposed Method: MetaPhaPred

We propose a novel deep learning model called MetaPhaPred, for metagenomic phage sequences prediction. The detailed architecture of MetaPhaPred is illustrated in Fig. 1. We first use dna2vec [28] to learn the distributed feature representations of k-mer words and encode the DNA sequences using the trained word vectors. Then, we feed the embedded sequences into CNN and Bi-LSTM to extract the feature maps and the long-term dependencies between the features. Next, the generated feature vector is then fed into an attention layer to enhance contributions of important features. The output is fed into a sigmoid unit to make a prediction. Further details of our framework are presented below.

1) Sequence Embedding: Most of the deep learning-based methods mainly used the one-hot encoding technique to represent DNA sequences. The distance between any two one-hot vectors is equal and the correlation information between nucleotides is ignored, which may affect the prediction accuracy. Therefore, we use the word embedding technique commonly utilized in natural language processing to learn the distributed representations of k-mers by applying dna2vec [28], which is based on the word2vec word embedding model [27]. The dna2vec first uses the one-hot encoding technique to encode the metagenomic sequences into one-hot vectors, and then adopts a two-layered neural network to learn the word vectors from the one-hot vectors. The architecture of dna2vec is shown in the upper part of Fig. 1.

Definition 1: k-mer. The k-mer w is composed of four nucleotide bases {A, G, C, T} with a length of k, which is extracted from the DNA sequence. Each k-mer can be considered as a word, and the vocabulary size is 4^k.

Definition 2: word sequence. The word sequence is composed of a set of ordered words, each of which is considered as a k-mer extracted from the DNA sequence under a control parameter, namely stride size q. For example, given a DNA sequence 'TGATGCAG' and $q = 1$, we can orderly transform this sequence as the word sequence 'TGA, GAT, ATG, TGC, GCA, CAG'.

Definition 3: one-hot vector. The one-hot vector uses a $\{0, 1\}$ vector v_w with size n_v to denote a word w in the word sequence, where n_v is the number of words. For example, given the word sequence 'TGA, GAT, ATG, TGC, GCA, CAG', the one-hot vector of 'GAT' can be denoted as [0 1 0 0 0 0]$^{\mathrm{T}}$, where [0 1 0 0 0 0]$^{\mathrm{T}}$ is the transposition of [0 1 0 0 0 0].

Definition 4: word vector. The word vector $\mathbf{v}$ is the learned representation of a two-layered neural network denoted by the output of the first layer of neural network with the one-hot vectors as the input.

Given a DNA sequence $x = (a_1, a_2, \ldots, a_m)$, dna2vec first splits this sequence into a word sequence with n k-mers, namely $w_1, w_2, \ldots, w_n$. Each k-mer w_t, $t = 1, 2, \ldots, n$, is represented as follows:

$$w_t = x_{k(t-1)+q} : x_{(kt+q)}, \quad t \in \{1, 2, \ldots, n\}, \tag{1}$$

where $x_{k(t-1)+q} : x_{(kt+q)}$ denotes a vector of elements in x coming from the $k(t-1)+q$ to $(kt+q)$ position. Then, w_t is encoded into a n_v-dimension one-hot vector v_{w_t} as follows:

$$v_{w_t} = [z_1^t, z_2^t, \ldots, z_{n_v}^t]^{\mathrm{T}}, \tag{2}$$

where n_v is the vocabulary size. For each $l \in \{1, 2, \ldots, n_v\}$, z_l^t is calculated as follows:

$$z_l^t = \begin{cases} 1, & \text{if } l \text{ is the index of the word } w_t \text{ in the vocabulary;} \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Next, the dna2vec uses a two-layered neural network to learn $\mathbf{v}$ from the above one-hot vectors. In this step, dna2vec follows the principle of the skip-gram algorithm [35], which uses the current word to maximally predict the surrounding words in a size of context window s. More specifically, we let the matrix $\mathrm{W}_{n_v \cdot n_d}$ and $\mathrm{W}'_{n_d \cdot n_v}$ be the weights of the first and second layer of the neural network, respectively. Given the one-hot vector of a k-mer w_t, the predicted one-hot probability vector $v'_{w_{(t+j)}}$ of its surrounding words (e.g., $w_{(t+j)}$) is computed as follows:

$$v'_{w_{(t+j)}} = \mathbf{softmax}(v_{w_t}^{\mathrm{T}} \cdot \mathrm{W} \cdot \mathrm{W}')^{\mathrm{T}}, \tag{4}$$

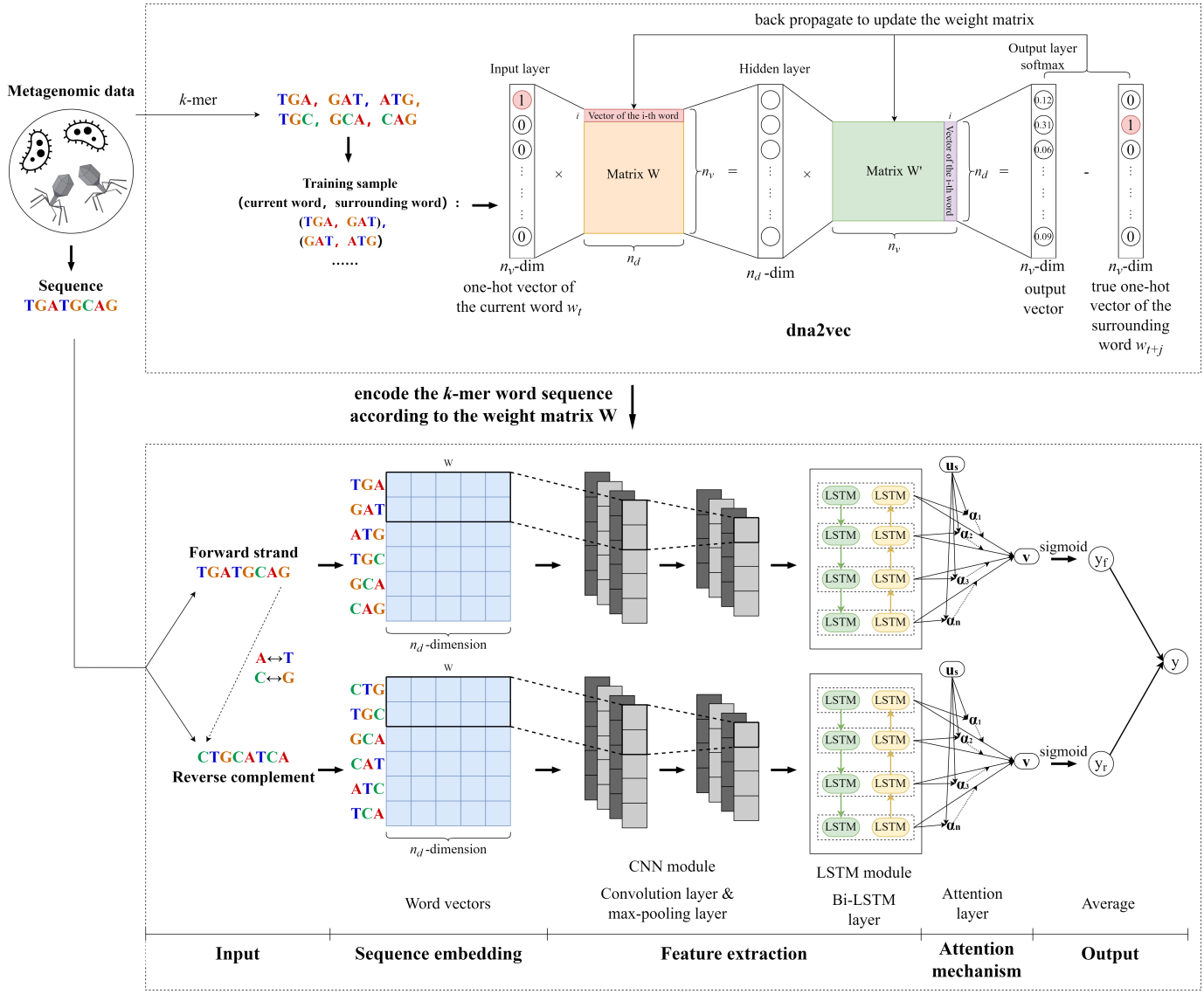


Fig. 1. The architecture of MetaPhaPred. MetaphaPred first uses a word embedding technique (dna2vec) to encode the metagenomic sequences into word vectors. The dna2vec first uses the one-hot encoding technique to encode the metagenomic sequences into one-hot vectors, and then adopts a two-layered neural network to learn the word vectors from the one-hot vectors. Then, MetaphaPred uses a CNN to capture the feature maps in sequences, and adopts Bi-LSTM to capture the long-term dependencies between features from both forward and backward directions. Next, an attention mechanism is used to enhance contributions of important features.

where $\mathbf{softmax}$ is the softmax function, and it is defined as follows:

$$\mathbf{softmax}(u_i) = \frac{e^{u_i}}{\sum_{t=1}^{n_v} e^{u_t}}, \tag{5}$$

where u_i is the value of ith index in $v^{'}_{w_{(t+j)}}$ and n_v is the vocabulary size.

In this case, dna2vec tries to minimize the differences between the predicted one-hot probability vector $v^{'}_{w_{(t+j)}}$ and the real one-hot vector $v_{w_{(t+j)}}$ for all surrounding words of all words (See Fig. 1). This minimization objective is equivalent to minimize the following conditional probability:

$$\begin{aligned} & -\frac{1}{n}\sum_{t=1}^{n}\sum_{-s\le j\le s, j\neq 0} \log\, p(w_{(t+j)}|w_t) \\ & = -\frac{1}{n}\sum_{t=1}^{n}\sum_{-s\le j\le s, j\neq 0} \log\, v^{'}_{w_{(t+j)}}(t+j), \end{aligned} \tag{6}$$

where $p(w_{(t+j)}|w_t)$ is the conditional probability of the appearance of word $w_{(t+j)}$ when word w_t is at the position t, and $v^{'}_{w_{(t+j)}}(t+j)$ is the value of $(t+j)$th index in $v^{'}_{w_{(t+j)}}$.

After training, the weight matrix W is obtained as the word vectors of all the n_v words in the vocabulary. The i-th row of the weight matrix W represents the n_d-dimension word vector of the i-th word in the vocabulary. More specifically, $\mathbf{v}_i = v_{w_t}^{\mathrm{T}} \cdot \mathrm{W} = \mathrm{W}(i,:)$, where $\mathrm{W}(i,:)$ is the ith row of W.

2) Feature Extraction: The feature patterns of nucleotide in a DNA sequence are usually represented by a position weight matrice (PWM) [19]. CNN is capable of extracting feature patterns using convolutional filters that slide along the input feature vectors of DNA words. Each convolutional filter is similar to a PWM and the convolution operation can be considered as scanning the PWM using a window that slide along the sequence [36], [37]. This characteristic of CNN enables it to help the phage prediction from metagenomic data. LSTM can effectively learn the long-term dependency between the feature patterns. We therefore combine CNN and LSTM to extract the feature patterns of phages and the long-term dependencies between the features in the sequences.

We first use the CNN to extract the feature maps of phages, namely a set of feature patterns. The CNN module consists of a one-dimensional convolution layer and a max-pooling layer. The convolution layer contains a set of convolution filters that slide along the input feature vectors learned from dna2vec. The convolution operation computes the inner product of the convolution kernels with the input feature vectors and generates the feature maps. Formally, a feature map $\mathbf{x}^C$ is generated by a convolutional filter with kernel size h as follows:

$$\begin{aligned} x_i^C &= f(W^C * \mathbf{V}_{i:i+h-1} + b^C), \\ \mathbf{V}_{i:i+h-1} &= \{\mathbf{v}_i, ..., \mathbf{v}_{(i+h-1)}\}, \\ \mathbf{v}_i &= \mathrm{W}(i,:), \\ \mathbf{x}^C &= [x_1^C, x_2^C, \ldots, x_{n-h+1}^C], \end{aligned} \tag{7}$$

where $\mathbf{v}_i$ is the word vector corresponding to the i-th word in the word sequence, while W is the weight matrix obtained in dna2vec. W^C and b^C are the trainable weight and bias of the convolutional filter, respectively. $*$ denotes the convolution operation and f is the activation function. x_i^C represents the value of i-th feature pattern and $\mathbf{x}^C$ represents the feature map.

Rectified Linear Unit (ReLU) [38] is set as the activation function to introduce non-linearity and prevent vanishing gradient, defined as:

$$\mathrm{ReLU}(u) = \max(0, u). \tag{8}$$

A max-pooling layer is used after the 1D convolution layer to subsample the feature maps by taking the maximum value over a sliding window. A dropout layer [39] is added after the max-pooling layer to prevent overfitting by randomly dropping units from the neural network.

Subsequently, the generated feature vector is fed into an bi-directional LSTM layer. LSTM is a variant of RNN that can solve the long-term dependency problem by introducing a gate mechanism. A LSTM cell consists of three gates that are used to control the cell states: namely a forget gate, an input gate, and an output gate, determining which information will be thrown away, kept, and output, respectively. Fig. 2 illustrates the structure of a LSTM cell. The calculation formulae of

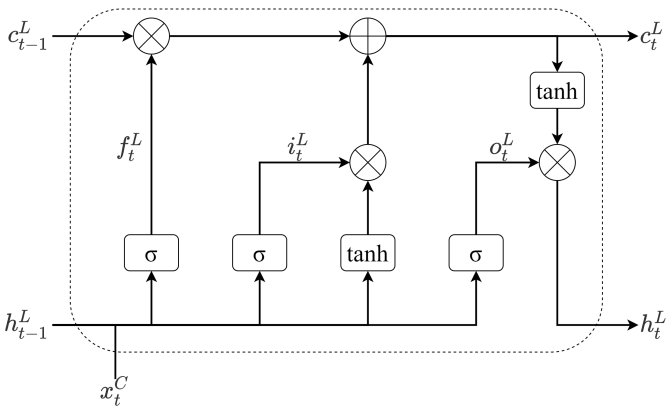


Fig. 2. The structure of a LSTM cell.

LSTM unit are as follows:

$$\begin{aligned} f_t^L &= \sigma(W_f^L x_t^C + U_f^L h_{t-1}^L + b_f^L), \\ i_t^L &= \sigma(W_i^L x_t^C + U_i^L h_{t-1}^L + b_i^L), \\ o_t^L &= \sigma(W_o^L x_t^C + U_o^L h_{t-1}^L + b_o^L), \\ c_t^L &= f_t^L \odot c_{t-1}^L + i_t^L \odot \tanh(W_c^L x_t^C + U_c^L h_{t-1}^L + b_c^L), \\ h_t^L &= o_t^L \odot \tanh(c_t^L), \end{aligned} \tag{9}$$

where $f_t^L, i_t^L, o_t^L, c_t^L$, and h_t^L represent the forget gate, input gate, output gate, cell state, and hidden state at position t, respectively. W_f^L, W_i^L, W_o^L, and W_c^L are the weights of x_t^C, while U_f^L, U_i^L, U_o^L, and U_c^L are the weights of h_{t-1}^L, b_f^L, b_i^L, b_o^L, and b_c^L are the biases, corresponding to forget gate, input gate, output gate, and memory cell, respectively. $\odot$ represents element-wise multiplication, while σ and $\tanh$ denote the sigmoid and hyperbolic tangent function, respectively.

Bi-directional LSTM (Bi-LSTM) [40] processes the input features from both forward and backward directions. The hidden state at position t is formalized as follows:

$$h_t^L = \overrightarrow{h_t^L} \oplus \overleftarrow{h_t^L}, \tag{10}$$

where $\overrightarrow{h_t^L}$ and $\overleftarrow{h_t^L}$ are the forward and backward hidden states at position t, respectively, while $\oplus$ is the concatenation operation. The Bi-LSTM can extract the long-term dependencies from both forward and backward directions.

3) Attention Mechanism: Certain potential features in the sequence are influential for phage prediction. Here, we use an attention mechanism [30] layer after the Bi-LSTM layer to learn the importance of all features and enhance contributions of key features. The formulae of the attention mechanism are as follows:

$$u_i^A = \tanh(W^A h_i^L + b^A), \tag{11}$$

$$\alpha_i = \frac{\exp(u_i^{A\top} u_s^A)}{\sum_i \exp(u_i^{A\top} u_s^A)}, \tag{12}$$

$$v^A = \sum_i \alpha_i h_i^L, \tag{13}$$

where h_i^L is the i-th feature from the Bi-LSTM layer, and u_i^A is a hidden representation of h_i^L. W^A and b^A are the weight

matrix and the bias, respectively, while $\tanh$ denotes the hyperbolic tangent function. The importance of each feature is measured by the similarity of u_i^A and a context vector u_s^A, while the normalized importance weight α_i is computed using the softmax function. The final output vector v^A is the sum of each feature vector multiplied by the corresponding importance weight.

4) Phage Prediction: The output of the attention layer is then fed into a sigmoid unit (σ) to compute a prediction score between 0 and 1, indicating the probability p that the given sequence is a phage, defined as:

$$\sigma(v^A) = \frac{1}{1 + \exp^{-v^A}}. \tag{14}$$

As a DNA sequence is double-stranded, both the forward sequence and the reverse complementary strand are input simultaneously to the same neural network to make a prediction. The final prediction score is the average of the two scores, defined as:

$$y = \frac{y_f + y_r}{2}, \tag{15}$$

where y_f and y_r are the prediction scores of the forward sequence and the reverse complementary strand, respectively. A similar technique was used in DeepVirFinder [19].

The models trained by sequences of length 100-400 bp, 401-800 bp, and 801-1200 bp are used to predict the sequences with corresponding length. When predicting sequences longer than 1200 bp, we first split the sequence into non-overlapping subsequences shorter than 1200 bp, and then use the corresponding model to predict each subsequence. The final prediction score of the sequence is the weighted average score of each subsequence. A similar techniques was used in PPR-meta [18].

D. Model Training

The binary cross-entropy loss function [41] was used to compute the error between the prediction scores and labels, defined as:

$$Loss = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log p_i + (1 - y_i)\log(1 - p_i)], \tag{16}$$

where y_i denotes the real label of x_i belonging to $\{0, 1\}$, representing bacteria/archaea and phage sequence respectively, and p_i denotes the prediction score of the sequence x_i. During the training process, the loss is back propagated to update the weights of the model. The Adam algorithm [42] is used as the optimizer to minimize this loss function.

We used the same hyperparameter settings for all datasets. Moreover, we adopted all sequences in the training sets as the corpus to train the word vectors. The training sets contain 4^k words (k-mers) at most, and each word (k-mer) is a combination of {A, G, C, T} with length k. In general, the k-mers in the test sets are also in the training sets when k is set as a small value. However, it is possible for that the k-mers are in the test sets but not in the training sets when k is set as a high value. In this case, similar to the processing method in [43], [44], we use a special word vector with all zeros to represent these words (k-mers). This processing method can effectively reduce the negative impacts of infrequent k-mers on the word representation. All parameters were manually fine-tuned and the optimal parameters setting are listed in Table III.

TABLE III
THE PARAMETERS OF THE PROPOSED MODEL.

Parameter	Value
k-mer length	3
Embedding stride size	1
Embedding size	100
Convolutional layer number	1
Convolutional filter number	512
Convolutional kernel size	40
Max-pooling window size	20
Max-pooling stride size	20
Dropout rate	0.5
Neurons of the Bi-LSTM layer	32
Neurons of the attention layer	32
Batch size	512
Learning rate	0.001
Epochs	15

The MetaPhaPred model is implemented in Python 3.8.8 based on Keras 2.8.0 with Nvidia RTX 3090 GPU.

E. Evaluation Metrics

We used four metrics to evaluate the prediction performance of our MetaPhaPred and all baseline methods, namely: Matthews correlation coefficient (MCC), accuracy (ACC), F1-score (F1), and area under the receiver operating characteristic curves (AUROC) [45].

The prediction made by a binary classifier can be divided into four categories: true positive (TP) and true negative (TN) are positive and negative examples correctly predicted, respectively, false positive (FP) are negative examples incorrectly predicted as positive, and false negative (FN) are positive examples incorrectly predicted as negative. The receiver operating characteristic curve shows the relationship between the true positive rate (TPR) and the false positive rate (FPR). Higher AUROC values indicate a better prediction performance of the model. The calculation formulae are as follows:

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}}, \tag{17}$$

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \tag{18}$$

$$\mathrm{TPR} = \mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{19}$$

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{TN} + \mathrm{FP}}, \tag{20}$$

$$\mathrm{F1} = \frac{2 \times \mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}, \tag{21}$$

$$\mathrm{MCC} = \frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN}}{\sqrt{(\mathrm{TP} + \mathrm{FP})(\mathrm{TP} + \mathrm{FN})(\mathrm{TN} + \mathrm{FP})(\mathrm{TN} + \mathrm{FN})}}. \tag{22}$$

This article has been accepted for publication in IEEE/ACM Transactions on Computational Biology and Bioinformatics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCBB.2023.3322870

F. Baseline Methods

Eleven state-of-the-art methods for metagenomic phage sequence identification were chosen as baseline methods to validate the effectiveness of the proposed method, including MetaPhinder [14], VirFinder [12], VIBRANT [16], VirSorter2 [17], PPR-meta [18], DeepVirFinder [19], VirNet [20], Seeker [21], RNN-VirSeeker [22], Virtifier [23], and PhaMer [24]. The reason for choosing MetaPhinder as baseline method is to show the limitation of sequence alignment-based methods. Comparisons of MetaPhaPred with VirFinder, VIBRANT, and VirSorter2 are made to validate the superiority of MetaPhaPred over the machine learning-based methods, while comparisons of MetaPhaPred with PPR-meta, DeepVirFinder, VirNet, Seeker, RNN-VirSeeker, Virtifier, and PhaMer are made to validate the superiority of the proposed deep neural network framework over the existing deep learning-based methods.

MetaPhinder [14]: It compares the contigs to the pre-built phage genome database using BLAST to identify phage contigs.

VirFinder [12]: It computes the k-mer frequency of the contigs and trains a logistic regression model to identify viral sequences.

VIBRANT [16]: It extracts protein features and uses a hybrid machine learning model for virus identification.

VirSorter2 [17]: It trains random forest classifiers based on the gene features to identify virus.

PPR-meta [18]: It designs a bi-path CNN to identify phage and plasmid sequences from the metagenomic data.

DeepVirFinder [19]: It uses a CNN to predict prokaryotic viral sequences.

VirNet [20]: It employs a deep attention model for viral identification of metagenomic data.

Seeker [21]: It trains a LSTM model for phage sequences identification.

RNN-VirSeeker [22]: It uses a LSTM to predict short viral sequences.

Virtifier [23]: It employs an attention-based LSTM for short viral sequences identification.

PhaMer [24]: It converts DNA sequences into protein-token sentences and designs a transformer-based model for phage identification.

III. Experimental Results

A. Performance Comparison on Test Datasets

We evaluated MetaPhaPred on the test sets with metagenomic sequences of different lengths. The normalized 2-class confusion matrices (shown in Fig. 3) are used to demonstrate the overall performance of MetaPhaPred. Specifically, the miss classification of phage and bacteria is below 18%, 13%, 10%, and 4% for sequences of length 100-400 bp, 401-800 bp, 801-1200 bp, and 5000-10000 bp, respectively. Moreover, MetaPhaPred has a good performance in the phage identification for metagenomic sequences with long sequences.

To demonstrate the effectiveness of MetaPhaPred, we compared it with several state-of-the-art methods (MetaPhinder [14], VirFinder [12], VIBRANT [16], VirSorter2 [17], PPR-meta [18], DeepVirFinder [19], VirNet [20], Seeker [21], RNN-VirSeeker [22], Virtifier [23], and PhaMer [24]). We tested the performance of MetaPhaPred and other baseline methods using the mentioned datasets and evaluation metrics in Section II. To facilitate fair comparison, for methods that support training a customized model, we retrained these methods with our training sets and reported the best results of the pre-trained models and the retrained models. Table IV summarizes the performance comparison for MetaPhaPred versus the baseline methods on the test sets.

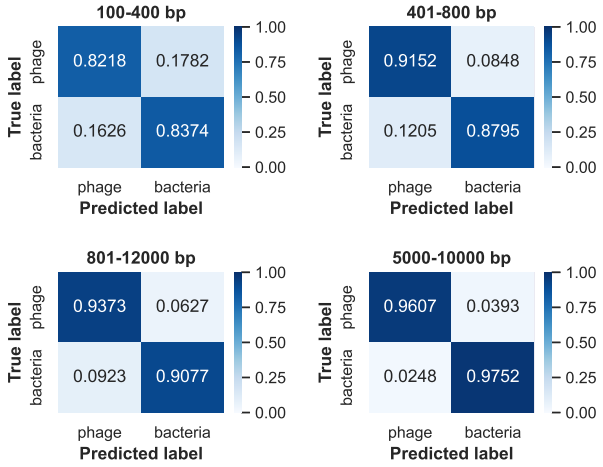


Fig. 3. The normalized 2-class confusion matrices of MetaPhaPred on the test sets with metagenomic sequences of different lengths.

The results in Table IV show that MetaPhaPred outperforms all baseline methods in all sequence lengths. All the tested methods exhibit a high prediction accuracy for long sequences. This is likely because long sequences contain sufficient informative features facilitating the classification. Notably, the sequence alignment-based method MetaPhinder obtain poor performance, especially for short sequences. In particular, MetaPhinder cannot predict sequences shorter than 500 bp. This may be because MetaPhinder relies on sequence alignment against the previously constructed phage gene databases and metagenomic sequences are too short to cover sufficient complete genes to facilitate prediction. Moreover, the deep learning-based methods (MetaPhaPred, PPR-meta, DeepVirFinder, VirNet, Seeker, RNN-VirSeeker, Virtifier, and PhaMer) outperform the machine learning-based methods (VirFinder, VIBRANT, and VirSorter2) in most cases, because they have advantages in extracting high-level features and information. The CNN-based methods PPR-meta and DeepVirFinder achieve the second-best and third-best performance in most cases, respectively. VirNet, Seeker, RNN-VirSeeker and Virtifier have relatively poor performance for long sequences, as the LSTM-based model with an overly long input sequence may result in the memory impairment. PhaMer converts DNA sequences into protein-token sentences. For phages that do not contain any tokens in the established vocabulary, PhaMer records them as false negative, which leads to relatively poor performance for short sequences. Note that, MetaPhaPred can predict metagenomic sequences with various lengths accurately, which cannot be addressed by most

TABLE IV
THE MCC, ACC, F1, AND AUROC PERFORMANCE COMPARISON OF METAPHAPRED AND THE BASELINE METHODS ON THE TEST SETS. THE BEST PERFORMANCE ARE HIGHLIGHTED IN BOLDFACE. "-" REPRESENTS THAT THE METRIC IS NOT AVAILABLE.

Datasets	Metrics	MetaPhaPred	MetaPhinder	VirFinder	VIBRANT	VirSorter2	PPR-meta	DeepVirFinder	VirNet	Seeker	RNN-VirSeeker	Virtifier	PhaMer
100-400 bp	MCC	**0.6592**	-	0.4930	-	0.0536	0.6402	0.5984	0.5008	0.3734	0.5723	0.5705	0.6284
	ACC	**0.8296**	-	0.7459	-	0.5029	0.8200	0.7989	0.7498	0.6867	0.7862	0.7797	0.7875
	F1	**0.8282**	-	0.7370	-	0.0114	0.8179	0.7942	0.7406	0.6891	0.7861	0.7557	0.7338
	AUROC	**0.9064**	-	0.8209	-	-	0.9039	0.8763	0.8365	0.7498	0.8731	0.8785	-
401-800 bp	MCC	**0.7952**	0.6781	0.6299	-	0.2539	0.7639	0.7400	0.6409	0.4373	0.6754	0.7213	0.7266
	ACC	**0.8974**	0.8176	0.8135	-	0.5621	0.8819	0.8700	0.8200	0.7186	0.8305	0.8572	0.8512
	F1	**0.8991**	0.8448	0.8043	-	0.2234	0.8807	0.8694	0.8149	0.7217	0.8111	0.8467	0.8294
	AUROC	**0.9593**	-	0.8945	-	-	0.9515	0.9363	0.9025	0.7924	0.9320	0.9367	-
801-1200 bp	MCC	**0.8454**	0.5859	0.6862	0.1530	0.4368	0.8037	0.8063	0.7002	0.4720	0.6901	0.7864	0.7741
	ACC	**0.9225**	0.7568	0.8415	0.5229	0.6666	0.9014	0.9030	0.8496	0.7360	0.8405	0.8910	0.8812
	F1	**0.9236**	0.8039	0.8334	0.0878	0.5075	0.8990	0.9015	0.8454	0.7383	0.8263	0.8849	0.8699
	AUROC	**0.9747**	-	0.9188	-	-	0.9608	0.9600	0.9256	0.8127	0.9274	0.9584	-
5000-10000 bp	MCC	**0.9360**	0.7991	0.8469	0.8985	0.9304	0.9079	0.9213	0.7328	0.5406	0.0603	0.8931	0.8437
	ACC	**0.9680**	0.8967	0.9216	0.9471	0.9645	0.9531	0.9602	0.8661	0.7702	0.5300	0.9448	0.9204
	F1	**0.9677**	0.9025	0.9177	0.9443	0.9635	0.9516	0.9592	0.8632	0.7739	0.5019	0.9423	0.9168
	AUROC	**0.9934**	-	0.9750	-	-	0.9871	0.9900	0.9405	0.8573	0.5445	0.9894	-

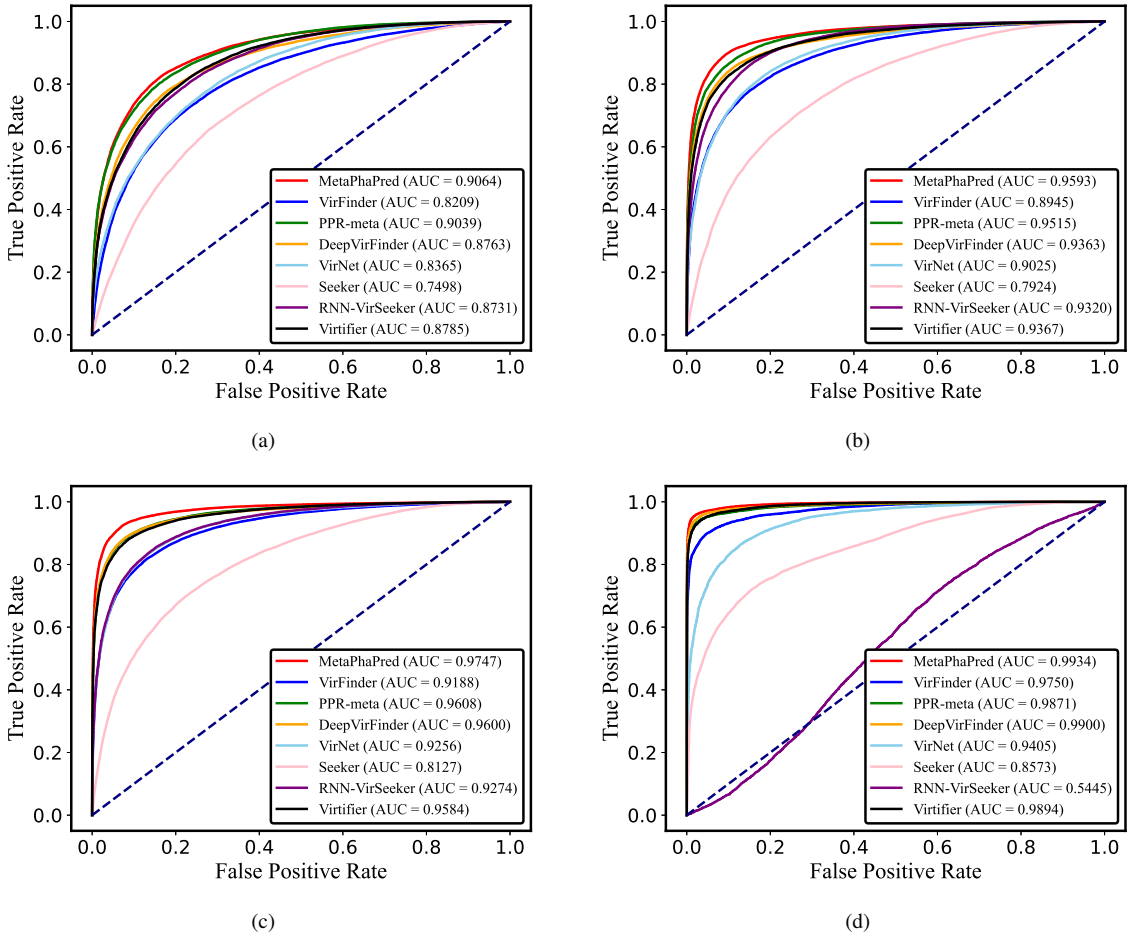


Fig. 4. The ROC curves of MetaPhaPred and the baseline methods on the test sets with lengths (a) 100-400 bp, (b) 401-800 bp, (c) 801-1200 bp, and (d) 5000-10000 bp.

of the baseline methods. The experimental results indicate that a combination of word embedding, CNN, LSTM, and attention mechanism has great advantages in predicting metagenomic phage sequences.

The ROC curves of MetaPhaPred and the baseline methods that have prediction scores are plotted in Fig. 4. The results show that MetaPhaPred obtains higher AUROC values than the other baseline methods at all sequence lengths. This further validates the superiority of the proposed MetaPhaPred over the baseline methods.

Moreover, we provided the average performance ranking of all methods to validate the overall performance. More

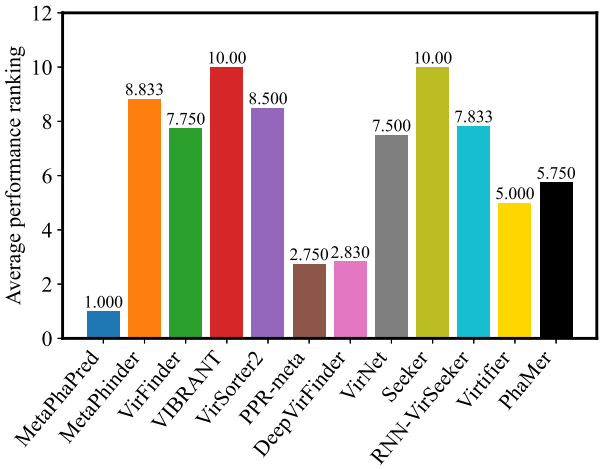


Fig. 5. Average performance ranking of MetaPhaPred and the baseline methods on the test sets.

specifically, all methods are ranked from 1 to 12 over all test sets, and the method with a better performance is ranked as a lower value. Fig. 5 plots the average performance ranking of all methods on the test sets. The results show that the average performance rank of MetaPhaPred (1.000) is much smaller than that of MetaPhinder (8.833), VirFinder (7.750), VIBRANT (10.000), VirSorter2 (8.500), PPR-meta (2.750), DeepVirFinder (2.830), VirNet (7.500), Seeker(10.000), RNN-VirSeeker (7.833), Virtifier (5.000), and PhaMer (5.750). This further validates the superiority of MetaPhaPred over the other baseline methods when comprehensively considering all the metrics on the test sets.

To further validate the effectiveness of the proposed method, we also compared MetaPhaPred with the methods that are the combination of the standard DNA sequence feature extraction tools such as the BioSeq-Analysis [46] and iLearn [47], and the classical machine learning algorithms such as the support vector machine (SVM) [48], random forest (RF) [49], logistic regression (LR) [12], k-nearest neighbours algorithm (KNN) [50], and naive bayes (NB) [51]. Here, BioSeq-Analysis and iLearn are used to calculate features for the DNA sequences, while the machine learning algorithms are adopted to sequence classification. Five representative DNA features such as the kmer [52], dinucleotide-based auto covariance (DAC) [53], trinucleotide-based auto covariance (TAC) [53], pseudo dinucleotide composition (PseDNC) [54], and pseudo k-tupler composition (PseKNC) [55] are extracted from the BioSeq-Analysis and iLearn toolkits. To facilitate fair comparisons, all these methods are trained using our in-house generated training sets. Table V summarizes the accuracy performance of all algorithms. In Table V, each baseline algorithm is named as the combination of the methods for extracting features of the DNA sequences and sequence classification. For example, the method Kmer-SVM is a combination of Kmer for extracting DNA features and SVM for sequence classification. The results show that MetaPhaPred achieved the best performance in all cases.

TABLE V
THE ACCURACY COMPARISON OF METAPHAPRED AND THE STANDARD DNA SEQUENCE FEATURE EXTRACTION TOOLS AND MACHINE LEARNING ALGORITHMS ON THE TEST SETS. THE BEST PERFORMANCE ARE HIGHLIGHTED IN BOLDFACE.

Methods	100-400 bp	401-800 bp	801-1200 bp	5000-10000 bp
Kmer-SVM [48], [52]	0.7195	0.7371	0.7478	0.7770
Kmer-RF [49], [52]	0.7359	0.7916	0.8287	0.8789
Kmer-LR [12], [52]	0.7165	0.7359	0.7462	0.7754
Kmer-KNN [50], [52]	0.6781	0.7542	0.7936	0.8229
Kmer-NB [51], [52]	0.7154	0.7240	0.7277	0.7204
DAC-SVM [48], [53]	0.7151	0.7327	0.7440	0.7714
DAC-RF [49], [53]	0.7027	0.7410	0.7633	0.8051
DAC-LR [12], [53]	0.7072	0.7282	0.7370	0.7695
DAC-KNN [50], [53]	0.6281	0.6873	0.7218	0.7174
DAC-NB [51], [53]	0.6925	0.7083	0.7168	0.7172
TAC-SVM [48], [53]	0.6574	0.6825	0.6997	0.7383
TAC-RF [49], [53]	0.6318	0.6697	0.6967	0.7571
TAC-LR [12], [53]	0.6485	0.6734	0.6910	0.7477
TAC-KNN [50], [53]	0.5735	0.6113	0.6445	0.6501
TAC-NB [51], [53]	0.6298	0.6587	0.6715	0.6900
PseDNC-SVM [48], [54]	0.7219	0.7413	0.7512	0.7755
PseDNC-RF [49], [54]	0.7361	0.7937	0.8303	0.8825
PseDNC-LR [12], [54]	0.7167	0.7362	0.7471	0.7753
PseDNC-KNN [50], [54]	0.6804	0.7566	0.7974	0.8253
PseDNC-NB [51], [54]	0.7137	0.7223	0.7262	0.7185
PseKNC-SVM [48], [55]	0.7329	0.7458	0.7492	0.7552
PseKNC-RF [49], [55]	0.7566	0.8166	0.8520	0.8925
PseKNC-LR [12], [55]	0.7253	0.7472	0.7625	0.7914
PseKNC-KNN [50], [55]	0.7231	0.7974	0.8346	0.8683
PseKNC-NB [51], [55]	0.7117	0.7159	0.7164	0.7109
MetaPhaPred	**0.8296**	**0.8974**	**0.9225**	**0.9680**

B. Performance Comparison on a Prophage Sequence Dataset

We tested the prophage identification ability of MetaPhaPred and the baseline methods. Table VI records the prophage recognition rate of each method on the prophage sequence dataset. The results show that MetaPhaPred outperforms all baseline methods in the ranges of 100-400 bp, 401-800 bp, and 801-1200 bp. In particular, compared with the second-best performance, MetaPhaPred achieves relative improvements of 3.95%, 3.06%, and 1.52% for prophage sequences with lengths 100-400 bp, 401-800 bp, and 801-1200 bp, respectively. For prophage sequences in the range of 5000-10000 bp, MetaPhaPred achieves the third-best performance, which is slightly lower than PhaMer and MetaPhinder. However, PhaMer and MetaPhinder have relatively poor performance for short prophage sequences.

C. Performance Comparison on a Real Metagenomic Phage Sequence Dataset

We evaluated the performance of MetaPhaPred and other baseline methods on the real metagenomic data. The metagenomic phage recognition rates of all methods on the real metagenomic phage sequence dataset are shown in Fig. 6. The results show that MetaPhaPred achieves the best performance. In particular, compared with the second-best performance, MetaPhaPred achieves a relative improvement of 2.08% for predicting the real metagenomic phage sequence.

D. Comparison of Running Time

We compared the running time of MetaPhaPred and all baseline methods. All the methods were run on Intel(R) Xeon(R) Silver 4215R CPU @ 3.20GHz with eight cores. Table VII shows the running time of predicting the real

TABLE VI
THE PROPHAGE RECOGNITION RATE COMPARISON OF METAPHAPRED AND THE BASELINE METHODS ON THE PROPHAGE SEQUENCE DATASET. THE BEST PERFORMANCE ARE HIGHLIGHTED IN BOLDFACE. "-" REPRESENTS THAT THE METRIC IS NOT AVAILABLE.

Datasets	MetaPhaPred	MetaPhinder	VirFinder	VIBRANT	VirSorter2	PPR-meta	DeepVirFinder	VirNet	Seeker	RNN-VirSeeker	Virtifier	PhaMer
100-400 bp	**0.6002**	-	0.4346	-	0.1672	0.5607	0.5603	0.4805	0.4448	0.5381	0.4252	0.4215
401-800 bp	**0.7018**	0.6574	0.4077	-	0.3243	0.6595	0.6712	0.5436	0.4280	0.4104	0.5237	0.5548
801-1200 bp	**0.7600**	0.7448	0.4194	0.0275	0.4619	0.7185	0.6667	0.5702	0.4280	0.5255	0.5628	0.6421
5000-10000 bp	0.7991	0.8013	0.4862	0.7192	0.7668	0.7820	0.7327	0.5971	0.4144	0.5266	0.6112	**0.8608**

TABLE VII
THE RUNNING TIME TO MAKE PREDICTIONS FOR THE REAL METAGENOMIC PHAGE SEQUENCE DATASET. THE BEST PERFORMANCE ARE HIGHLIGHTED IN BOLDFACE.

Method	MetaPhaPred	MetaPhinder	VirFinder	VIBRANT	VirSorter2	PPR-meta	DeepVirFinder	VirNet	Seeker	RNN-VirSeeker	Virtifier	PhaMer
Running time (min)	100	7	115	6	285	21	103	18	163	93	126	**3**

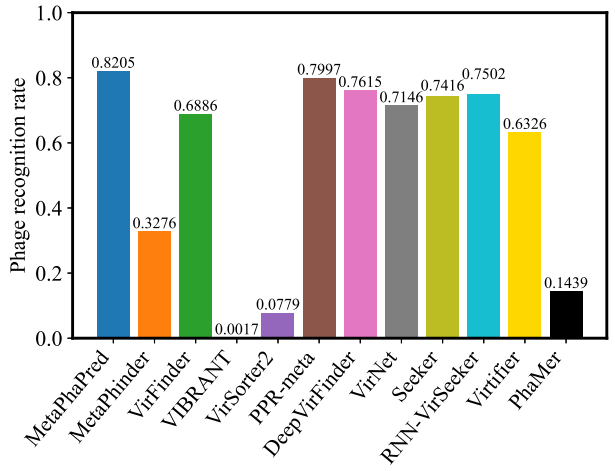


Fig. 6. The phage recognition rate comparison of MetaPhaPred and the baseline methods on the real metagenomic phage sequence dataset.

metagenomic phage sequence dataset for each method. Although it is not the fastest method, MetaPhaPred can obtain a good tradeoff between the prediction accuracy and the computational time.

E. Ablation Study

The above experimental comparison validates the effectiveness of MetaPhaPred, a novel deep learning model combined with word embedding, CNN, LSTM, and attention mechanism, for identifying phages from metagenomic data. To verify the effectiveness of each component of the model, we performed ablation study by removing each component and retraining the model, respectively. We consider the following variants of MetaPhaPred:

- MetaPhaPred-W: A variant of the MetaPhaPred model removing the embedding layer.
- MetaPhaPred-C: A variant of the MetaPhaPred model removing the CNN layer.
- MetaPhaPred-L: A variant of the MetaPhaPred model removing the Bi-LSTM layer.
- MetaPhaPred-A: A variant of the MetaPhaPred model removing the attention layer.

TABLE VIII
PERFORMANCE COMPARISON OF METAPHAPRED AND THE VARIANT MODELS. THE BEST PERFORMANCE ARE HIGHLIGHTED IN BOLDFACE. "METAPHAPRED" IS ABBREVIATED TO "M" IN THE TABLE.

Dataset	Metrics	M	M-W	M-C	M-L	M-A
100-400 bp	MCC	**0.6592**	0.5946	0.5664	0.6416	0.6420
	ACC	**0.8296**	0.7965	0.7828	0.8208	0.8207
	F1	**0.8282**	0.7886	0.7768	0.8215	0.8167
	AUROC	**0.9064**	0.8837	0.8678	0.9015	0.8994
401-800 bp	MCC	**0.7952**	0.7451	0.6947	0.7298	0.7889
	ACC	**0.8974**	0.8717	0.8470	0.8648	0.8944
	F1	**0.8991**	0.8759	0.8505	0.8666	0.8948
	AUROC	**0.9593**	0.9463	0.9258	0.9370	0.9560
801-1200 bp	MCC	**0.8454**	0.8083	0.7480	0.8243	0.8266
	ACC	**0.9225**	0.9041	0.8736	0.9122	0.9111
	F1	**0.9236**	0.9051	0.8706	0.9119	0.9063
	AUROC	**0.9747**	0.9620	0.9439	0.9680	0.9609
5000-10000 bp	MCC	**0.9360**	0.9169	0.8771	0.9274	0.9120
	ACC	**0.9680**	0.9585	0.9385	0.9635	0.9545
	F1	**0.9677**	0.9585	0.9389	0.9630	0.9526
	AUROC	**0.9934**	0.9901	0.9834	0.9927	0.9888

Table VIII illustrates the performance of MetaPhaPred and the variant models on the test sets. The experimental results show that MetaPhaPred outperforms all the variant models, which demonstrates the effectiveness of the word embedding, CNN, LSTM, and attention mechanism. More specifically, MetaPhaPred outperforms the variant models that removed the word embedding, CNN, LSTM, and attention mechanism at all sequence lengths, with a relative ACC improvement around 2%, 5%, 2%, and 1%, respectively.

F. Impacts of Parameter Settings

In this part, the impacts of certain key parameters on prediction performance of MetaPhaPred are analyzed. The simulated training and test sets were used in the experiments.

k denotes the length of word, which affects the learned sequence features of MetaPhaPred. Table IX shows the variations of the prediction performance of MetaPhaPred with different k values. The results illustrate that MetaPhaPred

TABLE IX
COMPARISON OF METAPHAPRED WITH DIFFERENT k-MER LENGTHS. THE BEST PERFORMANCE ARE HIGHLIGHTED IN BOLDFACE.

Datasets	Metrics	k=3	k=4	k=5	k=6
100-400 bp	MCC	0.6592	**0.6615**	0.6546	0.6434
	ACC	**0.8296**	0.8293	0.8267	0.8197
	F1	0.8282	0.8210	**0.8320**	0.8090
	AUROC	**0.9064**	0.9059	0.9051	0.8979
401-800 bp	MCC	0.7952	0.7859	0.7829	**0.7977**
	ACC	0.8974	0.8925	0.8914	**0.8986**
	F1	**0.8991**	0.8897	0.8923	0.8968
	AUROC	**0.9593**	0.9527	0.9530	0.9569
801-1200 bp	MCC	0.8454	**0.8492**	0.8444	0.8447
	ACC	0.9225	**0.9240**	0.9212	0.9215
	F1	**0.9236**	0.9219	0.9183	0.9190
	AUROC	**0.9747**	0.9696	0.9653	0.9695
5000-10000 bp	MCC	**0.9360**	0.9280	0.9254	0.9262
	ACC	**0.9680**	0.9634	0.9619	0.9623
	F1	**0.9677**	0.9623	0.9607	0.9611
	AUROC	**0.9934**	0.9922	0.9910	0.9924

TABLE X
COMPARISON OF METAPHAPRED WITH DIFFERENT NUMBER n_c OF CONVOLUTIONAL LAYERS. THE BEST PERFORMANCE ARE HIGHLIGHTED IN BOLDFACE.

Datasets	Metrics	$n_c = 1$	$n_c = 2$	$n_c = 3$	$n_c = 4$
100-400 bp	MCC	**0.6592**	0.6368	0.6227	0.6244
	ACC	**0.8296**	0.8174	0.8102	0.8121
	F1	**0.8282**	0.8243	0.8179	0.8141
	AUROC	**0.9064**	0.9003	0.8940	0.8939
401-800 bp	MCC	**0.7952**	0.7738	0.7541	0.7526
	ACC	**0.8974**	0.8865	0.8769	0.8763
	F1	**0.8991**	0.8836	0.8751	0.8761
	AUROC	**0.9593**	0.9538	0.9394	0.9427
801-1200 bp	MCC	**0.8454**	0.8347	0.8179	0.8266
	ACC	**0.9225**	0.9172	0.9085	0.9125
	F1	**0.9236**	0.9181	0.9062	0.9098
	AUROC	**0.9747**	0.9709	0.9663	0.9693
5000-10000 bp	MCC	**0.9360**	0.9245	0.9223	0.9230
	ACC	**0.9680**	0.9623	0.9608	0.9610
	F1	**0.9677**	0.9622	0.9599	0.9601
	AUROC	0.9934	**0.9939**	0.9924	0.9931

obtains the best performance in the phage prediction when $k = 3$ in most cases. This may be because a codon in biology is composed of three nucleotide bases, and the codon specifies the genetic information for synthesizing an amino acid and contains biological information. Moreover, these results illustrate that MetaPhaPred has a competitive performance for all k values. This validates that MetaPhaPred can effectively capture the potential features and correlations among the k-mers in the metagenomic data. In addition, Table IX shows that MetaPhaPred with high k values (i.e., $k = 5$ and $k = 6$) obtains a worse performance than that with low k values (i.e., $k = 3$ and $k = 4$) in most cases. This may be because a high k setting increases the growth of the vocabulary, and the k-mers are in the test sets but may be not in the training sets when k is set as a high value. In this case, k in MetaPhaPred is properly set as 3.

The convolutional layer plays an essential role in the prediction. We let n_c denote the number of convolutional layers, which contributes to the depth of CNNs and affects the weight training effects for prediction. Table X shows the variations of the prediction performance of MetaPhaPred with different n_c values. The results show that MetaPhaPred with $n_c = 1$ achieves the best performance at all sequence lengths, and it can obtain a competitive performance at all sequence lengths for all n_c values. The results also illustrate that the performance of MetaPhaPred decreases with the increase of n_c in most cases. This is probably because the increment of n_c increases the model complexity, namely the number of CNN weights, and the overfitting weight training would reduce the prediction performance of the model when the training sample is not enough. In this case, n_c in MetaPhaPred is properly set as 1.

The imbalance of datasets affects the performance of deep learning-based methods. Here, we test the the proposed MetaPhaPred on imbalanced datasets with different ratios of phage and bacteria sequences to validate its phage identification AUROC performance. The AUROC performance of MetaPhaPred is shown in Fig. 7. The results show that MetaPhaPred can always obtain high AUROC values for the imbalanced datasets with different ratios of phage and bacteria sequence. The results also show that the performance of MetaPhaPred is the best when the datasets are balanced, and the performance is degraded with the increase of imbalance. This is reasonable that the the deep learning-base methods are more likely to predict the minority class as the majority class incorrectly when the dataset is imbalance.

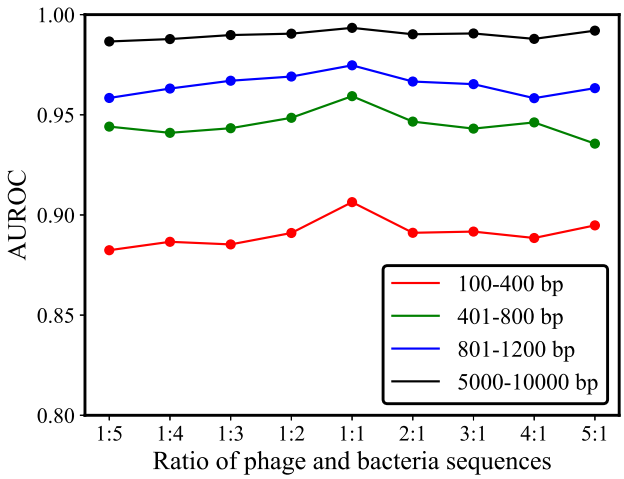


Fig. 7. The performance of MetaPhaPred on the simulated test sets with different ratios of phage and bacteria sequences.

IV. CONCLUSIONS

In this article, we have proposed a novel deep learning method, called MetaPhaPred, for identifying phage sequences from metagenomic data. In MetaPhaPred, we first employ the word embedding technique to learn the distributed fea-

ture representation of DNA words and use the trained word vectors to encode DNA sequences. This step can effectively extracts the latent feature vectors in the DNA words. Then, the embedded feature vectors are fed into CNN and Bi-LSTM to capture the feature maps, and the long-term dependence between the features from both forward and backward directions in sequences, respectively. Finally, we use an attention layer to enhance contributions of important features. The proposed MetaPhaPred has been tested on both the simulated and real metagenomic sequences with different lengths. The experimental results have demonstrated the superiority of our proposed MetaPhaPred in identifying metagenomic phage sequences with different lengths compared with several current state-of-the-art methods.

Note that, the proposed MetaPhaPred is limited for predicting sequences from other organisms such as plasmid and fungi. In future work, we will develop a deep learning-based multiple-classes classification model for various metagenomic data. Moreover, we will study the way for removing both the prophage and the homologous fragments of phages with the bacterial genomes, aiming to further improve the phage prediction accuracy of MetaPhaPred.

REFERENCES

[1] M. B. Dion, F. Oechslin, and S. Moineau, "Phage diversity, genomics and phylogeny," *Nature Reviews Microbiology*, vol. 18, no. 3, pp. 125–138, 2020.

[2] R. T. Schooley and S. Strathdee, "Treat phage like living antibiotics," *Nature Microbiology*, vol. 5, no. 3, pp. 391–392, 2020.

[3] G. P. C. Salmond and P. C. Fineran, "A century of the phage: past, present and future," *Nature Reviews Microbiology*, vol. 13, no. 12, pp. 777–786, 2015.

[4] F. Navarro and M. Muniesa, "Phages in the human body," *Frontiers in Microbiology*, vol. 8, p. 566, 2017.

[5] P. Manrique, M. Dills, and M. J. Young, "The human gut phage community and its implications for health and disease," *Viruses*, vol. 9, no. 6, p. 141, 2017.

[6] B. K. Chan, S. T. Abedon, and C. Loc-Carrillo, "Phage cocktails and the future of phage therapy," *Future Microbiology*, vol. 8, no. 6, pp. 769–783, 2013.

[7] S. Reardon, "Phage therapy gets revitalized: the rise of antibiotic resistance rekindles interest in a century-old virus treatment," *Nature*, vol. 510, no. 7503, pp. 15–17, 2014.

[8] M. Li *et al.*, "A deep learning-based method for identification of bacteriophage-host interaction," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 18, no. 5, pp. 1801–1810, 2021.

[9] L. Bragg and G. W. Tyson, "Metagenomics using next-generation sequencing," *Methods in Molecular Biology*, vol. 1096, pp. 183–201, 2014.

[10] D. H. Parks, M. Imelfort, C. T. Skennerton, P. Hugenholtz, and G. W. Tyson, "CheckM: assessing the quality of microbial genomes recovered from isolates, single cells, and metagenomes," *Genome Research*, vol. 25, no. 7, pp. 1043–1055, 2015.

[11] S. L. Smits *et al.*, "Assembly of viral genomes from metagenomes," *Frontiers in Microbiology*, vol. 5, p. 714, 2014.

[12] J. Ren, N. A. Ahlgren, Y. Y. Lu, J. A. Fuhrman, and F. Sun, "VirFinder: a novel k-mer based tool for identifying viral sequences from assembled metagenomic data," *Microbiome*, vol. 5, no. 1, p. 69, 2017.

[13] S. Roux, F. Enault, B. L. Hurwitz, and M. B. Sullivan, "VirSorter: mining viral signal from microbial genomic data," *PeerJ*, vol. 3, p. e985, 2015.

[14] V. I. Jurtz, J. Villarroel, O. Lund, M. Voldby Larsen, and M. Nielsen, "MetaPhinder—identifying bacteriophage sequences in metagenomic data sets," *PLoS ONE*, vol. 11, no. 9, p. e0163111, 2016.

[15] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *Journal of Molecular Biology*, vol. 215, no. 3, pp. 403–410, 1990.

[16] K. Kieft, Z. Zhou, and K. Anantharaman, "VIBRANT: automated recovery, annotation and curation of microbial viruses, and evaluation of viral community function from genomic sequences," *Microbiome*, vol. 8, p. 90, 2020.

[17] J. Guo *et al.*, "VirSorter2: a multi-classifier, expert-guided approach to detect diverse DNA and RNA viruses," *Microbiome*, vol. 9, p. 37, 2021.

[18] Z. Fang *et al.*, "PPR-Meta: a tool for identifying phages and plasmids from metagenomic fragments using deep learning," *GigaScience*, vol. 8, no. 6, p. giz066, 2019.

[19] J. Ren *et al.*, "Identifying viruses from metagenomic data using deep learning," *Quantitative Biology*, vol. 8, no. 1, pp. 64–77, 2020.

[20] A. O. Abdelkareem, M. I. Khalil, M. Elaraby, H. Abbas, and A. H. A. Elbehery, "VirNet: Deep attention model for viral reads identification," in *Proceedings of the 13th International Conference on Computer Engineering and Systems*, Cairo, Egypt, 2018, pp. 623–626.

[21] N. Auslander, A. B. Gussow, S. Benler, Y. I. Wolf, and E. V. Koonin, "Seeker: alignment-free identification of bacteriophage genomes by deep learning," *Nucleic Acids Research*, vol. 48, no. 21, p. e121, 2020.

[22] F. Liu, Y. Miao, Y. Liu, and T. Hou, "RNN-VirSeeker: A deep learning method for identification of short viral sequences from metagenomes," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 19, no. 3, pp. 1840–1849, 2022.

[23] Y. Miao, F. Liu, T. Hou, and Y. Liu, "Virtifier: a deep learning-based identifier for viral sequences from metagenomes," *Bioinformatics*, vol. 38, no. 5, pp. 1216–1222, 2022.

[24] J. Shang, X. Tang, R. Guo, and Y. Sun, "Accurate identification of bacteriophages from metagenomic data using Transformer," *Briefings in Bioinformatics*, vol. 23, no. 4, p. bbac258, 2022.

[25] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[27] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proceedings of International Conference on Learning Representations 2013*, Arizona, USA, 2013.

[28] P. Ng, "dna2vec: consistent vector representations of variable-length k-mers," *arXiv:1701.06279*, 2017.

[29] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2018*, Salt Lake City, USA, 2018.

[30] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, San Diego, USA, 2016, pp. 1480–1489.

[31] A. Vaswani *et al.*, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Long Beach, USA, 2017, pp. 6000–6010.

[32] S. Akhter, R. K. Aziz, and R. A. Edwards, "PhiSpy: a novel algorithm for finding prophages in bacterial genomes that combines similarity-and composition-based strategies," *Nucleic Acids Research*, vol. 40, no. 16, p. e126, 2012.

[33] S. Casjens, "Prophages and bacterial genomics: what have we learned so far?" *Molecular Microbiology*, vol. 49, no. 2, pp. 277–300, 2003.

[34] E. M. Ross, S. Petrovski, P. J. Moate, and B. J. Hayes, "Metagenomics of rumen bacteriophage from thirteen lactating dairy cattle," *BMC Microbiology*, vol. 13, p. 242, 2013.

[35] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems*, Lake Tahoe, USA, 2013, pp. 3111–3119.

[36] A. Trabelsi, M. Chaabane, and A. Ben-Hur, "Comprehensive evaluation of deep learning architectures for prediction of DNA/RNA sequence binding specificities," *Bioinformatics*, vol. 35, no. 14, pp. i269–i277, 2019.

[37] Q. Zhang, Z. Shen, and D.-S. Huang, "Predicting in-vitro transcription factor binding sites using DNA sequence + shape," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 18, no. 2, pp. 667–676, 2021.

[38] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning*, Madison, USA, 2010, pp. 807–814.

[39] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from over-

fitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.
[40] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, 2005.
[41] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of Operations Research*, vol. 134, no. 1, pp. 19–67, 2005.
[42] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proceedings of International Conference on Learning Representations 2015*, San Diego, USA, 2015.
[43] A. Subramanian, D. Pruthi, H. Jhamtani, T. Berg-Kirkpatrick, and E. Hovy, "Spine: Sparse interpretable neural embeddings," in *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, New Orleans, USA, 2018.
[44] C. Tan, F. Wei, N. Yang, B. Du, W. Lv, and M. Zhou, "S-net: From answer extraction to answer synthesis for machine reading comprehension," in *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, New Orleans, USA, 2018.
[45] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (ROC) curve." *Radiology*, vol. 143, no. 1, pp. 29–36, 1982.
[46] B. Liu, "BioSeq-Analysis: a platform for DNA, RNA and protein sequence analysis based on machine learning approaches," *Briefings in Bioinformatics*, vol. 20, no. 4, pp. 1280–1294, 2019.
[47] Z. Chen *et al.*, "iLearn: an integrated platform and meta-learner for feature engineering, machine-learning analysis and modeling of DNA, RNA and protein sequence data," *Briefings in Bioinformatics*, vol. 21, no. 3, pp. 1047–1057, 2020.
[48] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.
[49] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.
[50] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.
[51] J. D. Rennie, L. Shih, J. Teevan, and D. R. Karger, "Tackling the poor assumptions of naive bayes text classifiers," in *Proceedings of the 20th International Conference on Machine Learning*, Washington, D.C., USA, 2003, pp. 616–623.
[52] B. Liu, R. Long, and K.-C. Chou, "iDHS-EL: identifying DNase I hypersensitive sites by fusing three different modes of pseudo nucleotide composition into an ensemble learning framework," *Bioinformatics*, vol. 32, no. 16, pp. 2411–2418, 2016.
[53] Q. Dong, S. Zhou, and J. Guan, "A new taxonomy-based protein fold recognition approach based on autocross-covariance transformation," *Bioinformatics*, vol. 25, no. 20, pp. 2655–2662, 2009.
[54] W. Chen, P.-M. Feng, H. Lin, and K.-C. Chou, "iRSpot-PseDNC: identify recombination spots with pseudo dinucleotide composition," *Nucleic Acids Research*, vol. 41, no. 6, pp. e68–e68, 2013.
[55] S.-H. Guo *et al.*, "iNuc-PseKNC: a sequence-based predictor for predicting nucleosome positioning in genomes with pseudo k-tuple nucleotide composition," *Bioinformatics*, vol. 30, no. 11, pp. 1522–1529, 2014.

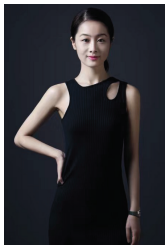
Lijia Ma received the B.S. degree in communication engineering from Hunan Normal University, Changsha, China, and the Ph.D. degree in electronic science and technology from Xidian University, Xi'an, China, in 2010 and 2015, respectively.

From 2015 to 2016, he was a Postdoctoral Fellow with Hong Kong Baptist University, Hong Kong, and with Nanyang Technological University, Singapore, from 2016 to 2017. He is an associate professor at the College of Computer and Software Engineering of Shenzhen University. His research interests mainly include evolutionary computation, machine learning and complex networks.

Wenwei Deng received the B.S. degree in biotechnology and the M.S. degree in computer technology from Shenzhen University, China, in 2020 and 2023, respectively. His research interests include bioinformatics and deep learning.

Yuan Bai received her PhD degree in computer software and theory from Jilin University, Changchun, China, in 2017. Then, she was a Postdoctoral Fellow at Hong Kong University, Hong Kong, China. Her research interests mainly focus on network epidemiology and machine learning.

Zhanwei Du is an assistant professor (research) at the School of Public Health at the University of Hong Kong (HKU). He received his PhD in computer architecture from Jilin University in 2015. Before joining HKU, he was a postdoc fellow at the Hong Kong Baptist University during 2015-2016, the University of Texas at Austin during 2016-2020, and a research associate at the University of Texas at Austin in 2020. His general research area is computational epidemiology, machine learning, and data science.



Minfeng Xiao received her PhD degree in 2013 from the University of Hong Kong. She is currently the associate director of Infection Omics Research Center and an investigator at BGI Research, BGI-Shenzhen. Her research interests include microbial genomics and synthetic biology.

Lin Wang received the B.S. degree from Southeast University, Nanjing, China, M.S. degree from Nankai University, Tianjin, China, and the Ph.D. degree from Fudan University, Shanghai, China, in 2006, 2009, and 2013, respectively. He was a chargé de recherché with the Institut Pasteur, France, from 2018 to 2020, and a postdoctoral fellow with the University of Hong Kong, Hong Kong, from 2014 to 2017. He is currently a senior research fellow at the Department of Genetics, University of Cambridge, UK. He is an elected member of the Global Young Academy. He has (co-)authored over 80 research publications in peer-reviewed journals. His research interests include infectious disease dynamics, immune dynamics, serology, pathogen genomics, and methodological advancements in Bayesian inference, machine learning, and the integration of heterogeneous data sources. He was a recipient of the Excellent Reviewer Award of the IEEE Transactions on Network Science and Engineering in 2022, Distinguished Referee Award from the European Physical Society in 2020, Best Talk Award at the 2018 International Conference on Molecular Epidemiology and Evolutionary Genetics, and the Outstanding Doctoral Dissertation Award of Shanghai Municipality in 2015. He serves as an Associate Editor for the BMC Medicine and Frontiers in Microbiology, an Editorial Board Member for BMC Infectious Diseases, and a Guest Editor for PLoS Neglected Tropical Diseases.

Jianqiang Li received his B.S. and Ph.D. Degree in automation major from South China University of Technology, Guangzhou, China, in 2003 and 2008, respectively.

He is a professor at the College of Computer and Software Engineering of Shenzhen University. He led three projects of the National Natural Science Foundation of China. His major research interests include embedded systems and Internet of Things.

Asoke K. Nandi (F'11) received Ph.D. degree in Physics from the University of Cambridge (Trinity College), Cambridge (UK). He held academic positions in several universities, including Oxford (UK), Imperial College London (UK), Strathclyde (UK), and Liverpool (UK) as well as Finland Distinguished Professorship in Jyvaskyla (Finland). In 2013, he moved to Brunel University London (UK), to become the Chair and Head of Electronic and Computer Engineering. Professor Nandi is a Distinguished Visiting Professor at Shenzhen University (China) and an Adjunct Professor at University of Calgary (Canada).

In 1983, Professor Nandi co-discovered the three fundamental particles known as W^+, W^- and Z^0 (by the UA1 team at CERN), providing the evidence for the unification of the electromagnetic and weak forces, for which the Nobel Committee for Physics in 1984 awarded the prize to his two team leaders for their decisive contributions. His current research interests lie in signal processing and machine learning, with applications to communications, image segmentations, biomedical data, etc. He has made many fundamental theoretical and algorithmic contributions to many aspects of signal processing and machine learning. He has much expertise in Big and Heterogeneous Data, dealing with modelling, classification, estimation, and prediction.

He has authored over 600 technical publications, including 250 journal papers as well as five books, entitled Condition Monitoring with Vibration Signals: Compressive Sampling and Learning Algorithms for Rotating Machines (Wiley, 2020), Automatic Modulation Classification: Principles, Algorithms and Applications (Wiley, 2015), Integrative Cluster Analysis in Bioinformatics (Wiley, 2015), Blind Estimation Using Higher-Order Statistics (Springer, 1999), and Automatic Modulation Recognition of Communications Signals (Springer, 1996). The H-index of his publications is 80 (Google Scholar) and his ERDOS number is 2.

Professor Nandi is a Fellow of the Royal Academy of Engineering (UK) as well as a Fellow of seven other institutions, including the IEEE and the IET. Among the many awards he received are the Institute of Electrical and Electronics Engineers (USA) Heinrich Hertz Award in 2012, the Glory of Bengal Award for his outstanding achievements in scientific research in 2010, the Water Arbitration Prize of the Institution of Mechanical Engineers (UK) in 1999, and the Mountbatten Premium, Division Award of the Electronics and Communications Division, of the Institution of Electrical Engineers (UK) in 1998. Professor Nandi is an IEEE EMBS Distinguished Lecturer (2018-19).