


Article

Enhancer-LSTMAtt: A Bi-LSTM and Attention-Based Deep Learning Method for Enhancer Recognition

Guohua Huang ^{1,*}, Wei Luo ¹, Guiyang Zhang ¹, Peijie Zheng ¹, Yuhua Yao ², Jianyi Lyu ¹, Yuewu Liu ³ and Dong-Qing Wei ⁴ 

¹ School of Electrical Engineering, Shaoyang University, Shaoyang 422000, China; qq25822@163.com (W.L.); guiyang9542@163.com (G.Z.); zhengpeijie1997@163.com (P.Z.); lyy990309@163.com (J.L.)

² School of Mathematics and Statistics, Hainan Normal University, Haikou 571158, China; yaoyuhua2288@163.com

³ College of Information and Intelligence, Hunan Agricultural University, Changsha 410083, China; yuewuliu@whu.edu.cn

⁴ State Key Laboratory of Microbial Metabolism, and School of Life Sciences and Biotechnology, Shanghai Jiao Tong University, Shanghai 200240, China; dqwei@sjtu.edu.cn

* Correspondence: 3280@hnsyu.edu.cn

Abstract: Enhancers are short DNA segments that play a key role in biological processes, such as accelerating transcription of target genes. Since the enhancer resides anywhere in a genome sequence, it is difficult to precisely identify enhancers. We presented a bi-directional long-short term memory (Bi-LSTM) and attention-based deep learning method (Enhancer-LSTMAtt) for enhancer recognition. Enhancer-LSTMAtt is an end-to-end deep learning model that consists mainly of deep residual neural network, Bi-LSTM, and feed-forward attention. We extensively compared the Enhancer-LSTMAtt with 19 state-of-the-art methods by 5-fold cross validation, 10-fold cross validation and independent test. Enhancer-LSTMAtt achieved competitive performances, especially in the independent test. We realized Enhancer-LSTMAtt into a user-friendly web application. Enhancer-LSTMAtt is applicable not only to recognizing enhancers, but also to distinguishing strong enhancer from weak enhancers. Enhancer-LSTMAtt is believed to become a promising tool for identifying enhancers.

Keywords: enhancer; promoter; deep learning; feed-forward attention; convolution neural network; long-short term memory; residual neural network



Citation: Huang, G.; Luo, W.; Zhang, G.; Zheng, P.; Yao, Y.; Lyu, J.; Liu, Y.; Wei, D.-Q. Enhancer-LSTMAtt: A Bi-LSTM and Attention-Based Deep Learning Method for Enhancer Recognition. *Biomolecules* **2022**, *12*, 995. <https://doi.org/10.3390/biom12070995>

Academic Editors: Xin Lai and Le Zhang

Received: 31 May 2022

Accepted: 7 July 2022

Published: 17 July 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Enhancers are short pieces of DNA sequences of 50 to 1500 bp, which can accelerate the transcription of target genes by binding the transcription factors [1,2]. Unlike the promoters, enhancers are located either in the upstream/downstream or within the genes they regulate and doesn't have to be close to the starting sites of transcription [2–4]. Increasing evidences indicate that enhancers play a critical role in the gene regulation [4,5]. The enhancers control the expression of genes involved in cell differentiation [6,7] and are responsible for morphological changes in three spine stickleback fish [8]. The enhancers orchestrate critical cellular events such as differentiation [9,10], maintenance of cell identity [11,12], and response to stimuli [13–15] by binding to transcription factors [16]. The enhancers are closely related to inflammation and cancer [17]. Therefore, precisely detecting enhancers from DNA sequences is critical to further investigate their functions or roles in the cellular processes.

The methods or techniques used to identify enhancers are divided into two categories: high-throughput experimental technology and computational method [5,18]. The former includes chromatin immunoprecipitation followed by deep sequencing (ChIP-seq) [19,20], protein-binding microarrays (PBMs) [21], systematic evolution of ligands by exponential enrichment (SELEX) [22], yeast-one-hybrid (Y1H) [23], and bacterial-one-hybrid [24]. The

main idea behind these technologies is to identify enhancers by recognizing properties of the enhancer-binding interactors [16]. There are generally four ways of experimental technology. The first way is to identify enhancers by binding sites of the specific transcription factors (TFs) with the help of ChIP-seq [13,25]. These techniques are restricted to cell-type or tissue-specific TFs. The second way is to detect enhancers by recognizing the binding sites of transcriptional co-activators such as CBP (also known as CREB-binding protein or CREBBP) and P300 (also called EP300 or E1A binding protein p300) recruited by the TFs [12,13,26]. However, not all enhancers are characterized by the co-activators, and the ChIP-grade antibodies are not always available. The third way is to identify nucleosome-depleted open regions of DNase I hypersensitivity [27]. The open regions include other DNA elements, such as promoters, silencers/repressors, insulators, and other function-unknown sequences [28,29]. The modifications of histones in the flanking nucleosomes are of certain signature of enhancers. For example, histones flanking active enhancers are typically marked by H3 mono-methylated at lysine 4 (H3K4me1), while histone flanking active promoters are marked by H3K4me3 [13]. Therefore, the fourth method is genome-wide mapping of histone modifications. In spite of great success in identifying enhancers, high-throughput experimental technologies have two drawbacks: they are time-consuming and expensive. Therefore, it is a challenging task to identify all enhancers from thousands of tissues or cells.

The computational methods have been developed to complement the high-throughput experimental technologies over the recent decade [18,30,31]. The computational methods include genomics comparison-based methods and machine learning-based methods. The enhancers reside in any region of the genome, so it is very difficult to find intuitively linear motifs of enhancers by genomics comparison-based methods. Machine learning-based methods build a classification model to fit known enhancers and then predict new enhancers. Furthermore, Machine learning-based methods are capable of discovering non-linear hidden motifs of enhancers. To date, there are at least twenty machine learning-based methods for enhancer prediction [16,32–63], such as iEnhancer-2L [40], iEnhancer-PseudeKNC [41], EnhancerPred [42], and EnhancerPred2.0 [43]. The general workflow of these methods is firstly to compute representation of sequences such as pseudo k-tuple nucleotide composition, nucleotide binary profiles, as well as accumulated nucleotide frequency, then to learn a classifier by using a machine learning algorithm such as support vector machine and random forest, and finally to predict unknown sequences.

The aforementioned machine learning-based methods require sophisticated design of representations as well as sophisticated selection of conventional machine learning algorithms. In practice, any single representation is not able to characterize enhancers well, while a combination of diverse representations has the potential to improve the performance but reduces the generalization ability of the methods. The deep learning methods that have been developed in the recent decades have proven to be good at addressing complex issues, including protein structure prediction, which is thought to be one of the most challenging tasks [64,65]. Yao et al. [60] presented a word embedding-based deep learning method named iEnhancer-GAN to detect enhancers. To make up for insufficiency of the number of training samples, iEnhancer-GAN [60] used the sequence generative adversarial net [66] to augment training samples. Min et al. [33] developed a deep convolution neural network (CNN)-based method for distinguishing enhancers from non-enhancers, which required only primary sequences as input. Khanal et al. [52] exploited word embedding in the field of natural language processing as well as CNN to construct a method named iEnhancer-CNN. Nguyen et al. [50] integrated multiple CNNs into the iEnhancer-ECNN. The CNN is capable of characterizing local properties [67], but is insufficient to represent semantic relationships between words in the context of sequences. Tan et al. [48] exploited recurrent neural networks (RNN) and integrated the output of both RNN and CNN for the final decision. Le et al. [55] presented an advanced method (BERT [68]) to capture semantics of DNA sequences. On the basis of analysis of the published works or methods for detecting

enhancers, we presented a bi-directional long-short term memory (Bi-LSTM) and attention-based deep learning method for enhancer recognition called Enhancer-LSTMAtt.

2. Data

For fair comparison with the state-of-the-art methods, we used the same benchmark dataset as those in iEnhancer-2L [40], iEnhancer-PsedeKNC [41], EnhancerPred [42], EnhancerPred2.0 [43], Enhancer-Tri-N [44], iEnhancer-2L-Hybrid [45], iEnhancer-EL [46], iEnhancer-5Step [47], DeployEnhancer [48], ES-ARCNN [49], iEnhancer-ECNN [50], EnhancerP-2L [51], iEnhancer-CNN [52], iEnhancer-XG [53], Enhancer-DRRNN [54], Enhancer-BERT [55], iEnhancer-KL [56], iEnhancer-RF [57], spEnhancer [58], iEnhancer-EBLSTM [59], iEnhancer-GAN [60], piEnPred [61], iEnhancer-RD [62], and iEnhancer-MFGBDT [63]. The dataset was initially collected by Liu et al. [40] from chromatin state information of nine cell lines (H1ES, K562, GM12878, HepG2, HUVEC, HSMM, NHLF, NHEK and HME) which was annotated by ChromHMM [69,70]. The initial enhancers included sequences of less than 200 bp and were of high homologies. In order to comply with the length of nucleosome and linker DNA, less than 200 bp sequences were removed and more than 200 bp sequences were cut into segments of fixed length (200 bp). Liu et al. [40] employed the CD-HIT to decrease or remove homologies among sequences. The CD-HIT [71–73] is a clustering tool to reduce redundant sequences. The generated non-redundant sequences were used to examine the dependency of methods on homology [74–80]. Liu et al. [40] set sequence identity to 0.8, indicating that homologies between chosen enhancers were not less than 0.8. The enhancers were grouped into strong enhancers and weak enhancers. The numbers of weak enhancers and the non-enhancers are much greater than those of strong enhancers. To achieve a balance between positive and negative samples, Liu et al. [40] randomly sampled the same numbers of weak enhancers as the strong enhancers and the same number of non-enhancers as the sum of strong and weak enhancers. The benchmark dataset S consisted of the strong enhancer set S_{strong} , the weak enhancer set S_{weak} , and the non-enhancer set S_{non} , whose numbers were 742, 742, and 1484, respectively. During the process of distinguishing the enhancer from the non-enhancer, both the strong enhancers and the weak enhancers were viewed as positive samples, and the non-enhancers were viewed as negative samples. During the process of distinguishing strong enhancers from weak enhancers, strong enhancers were positive, and weak enhancers were negative.

Another dataset S^i was used for the independent test, which was from reference [46]. The S^i contained 100 strong enhancers S_{strong}^i , 100 weak enhancers S_{weak}^i and 100 non-enhancers S_{non}^i . The sequence identities between any two enhancers are not more than 0.8 by processing by CD-HIT [71–73].

3. Methods

As shown in Figure 1, the proposed method comprised mainly input, embedding, 1D CNN, residual neural network (ResNet), Bi-LSTM, attention, dropout, flattened, and fully connected layers. The input was DNA segments of 200 bp. Then, DNA segments were transformed into number sequences by

$$f(X) = \begin{cases} 0 & A \\ 1 & C \\ 2 & G \\ 3 & T \\ 4 & N \end{cases} \quad (1)$$

where N denoted the characters of the unknown nucleotide. The embedding of number sequences was entered into the convolution module and the LSTM module. The convolution module consisted mainly of the 1D CNN and ResNet [81–83], while the LSTM module comprised mainly Bi-LSTM [84,85] and feed-forward attention [86,87]. The concatenation

of outputs of the two modules was entered into the fully connected layer. Following the fully connected layer was the final layer, which contained one neuron representing the probabilities of belonging to enhancers. We set the threshold to 0.5, and thus more than 0.5 output indicated that the corresponding input was predicted to be positive and otherwise to be negative. The numbers of the parameters and the shape of output in each layer of the Enhancer-LSTMAtt were listed in Table 1.

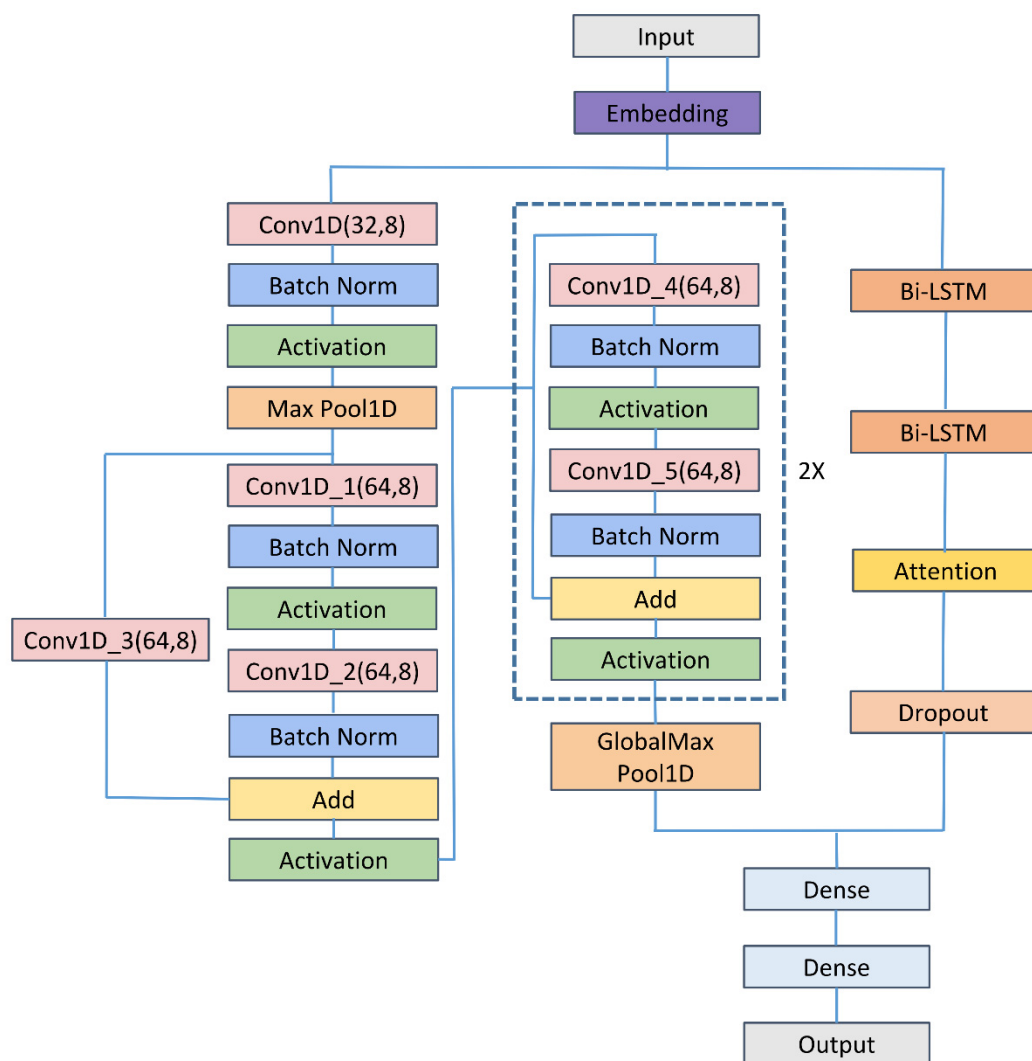


Figure 1. The architecture of the Enhancer-LSTMAtt. Conv1D, Batch Norm, Attention, Activation, Dense, and Max Pool 1D denote the 1D CNN layer, the batch normalization layer, the feed-forward attention layer, activation function, the fully-connected layer, and the max pooling layer respectively.

3.1. Embedding Layer

The embedding is generally the first layer of the deep neural network, whose role is to map the categorical (discrete) variable to continuous vectors (<https://towardsdatascience.com/neural-network-embeddings-explained-4d028e6f0526> (accessed on 3 March 2022)) [88]. The traditional one-hot encoding suffered from two defaults. One was that it was not capable of distinguishing similarities between representations. Another was that the representation was sparse in the case of the large vocabulary. The embedding well solved two issues and thus was widely applied to the area of natural language processing. The embedding can be used alone, such as word2vec and Glove, or fused into the deep neural network as the first layer.

Table 1. The shapes of outputs and the numbers of parameters in the Enhancer-LSTMAtt.

Layers	Shape of Output	Number of Parameters
Input	(None, 200)	0
Embedding	(None, 200, 32)	160
Conv1D(32, 8)	(None, 200, 32)	8224
Batch Normalization	(None, 200, 32)	128
Activation	(None, 200, 32)	0
Max Pooling	(None, 199, 32)	0
Conv1D_1(64, 8)	(None, 96, 64)	16,448
Batch Normalization	(None, 96, 64)	256
Activation	(None, 96, 64)	0
Conv1D_2(64, 8)	(None, 96, 64)	32,832
Batch Normalization	(None, 96, 64)	256
Conv1D_3(64, 8)	(None, 96, 64)	16,448
Add	(None, 96, 64)	0
Activation	(None, 96, 64)	0
Conv1D_4(64, 8)	(None, 96, 64)	32,832
Batch Normalization	(None, 96, 64)	256
Activation	(None, 96, 64)	0
Conv1D_5(64, 8)	(None, 96, 64)	32,832
Batch Normalization	(None, 96, 64)	256
Add	(None, 96, 64)	0
Activation	(None, 96, 64)	0
Conv1D_6(64, 8)	(None, 96, 64)	32,832
Batch Normalization	(None, 96, 64)	256
Activation	(None, 96, 64)	0
Conv1D_7(64, 8)	(None, 96, 64)	32,832
Batch Normalization	(None, 96, 64)	256
Add	(None, 96, 64)	0
Activation	(None, 96, 64)	0
Global Max Pooling	(None, 64)	0
Bidirectional LSTM	(None, 200, 64)	16,640
Bidirectional LSTM	(None, 200, 64)	24,832
Attention	(None, 64)	264
Dropout	(None, 64)	0
Concatenate	(None, 128)	0
Dense(16)	(None, 16)	2064
Dense(1)	(None, 1)	17

3.2. CNN

CNN is one of most popular neural network architectures used to construct deep neural network [67,89,90]. The main characteristic of the CNN is to capture the local hidden structure by using convolutional kernels or filters. As shown in Figure 2A, the input is divided into patches, which are convoluted into the feature map by the convolutional kernel. The patches are allowed to overlap, and the interval between adjacent patches is called the stride. All of the patches in the same input share the convolutional kernel which are learnable parameters. To keep the size of the input unchanged, the input is sometimes required to pad. To increase the non-linear ability of the CNN, the activation function is added to the feature map. The activation function includes ReLU, sigmoid, tanh, weakly ReLU, and ELU. The pooling in the CNN is a non-linear down-sampling, whose role is to reduce the dimensionality of representations and to speed up the calculation. In addition, the pooling is able to avoid or decrease the over-fitting issue.

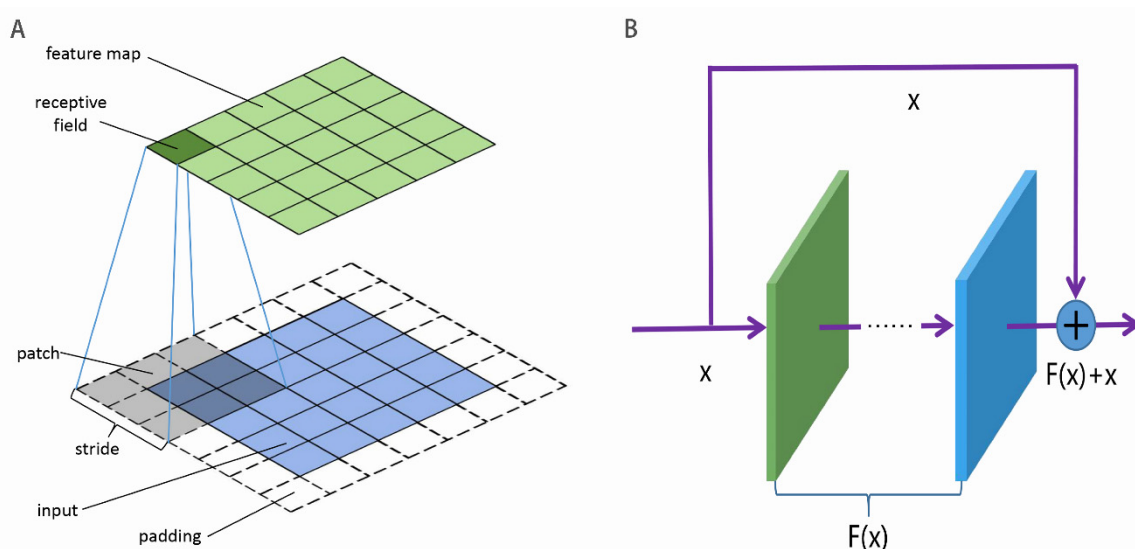


Figure 2. Illustration of (A) convolutions in the CNN and (B) the ResNet unit. $F(x)$ generally is CNN.

3.3. ResNet

As the number of stacked layers in the deep neural network increased, three issues would occur: information loss, gradient vanishing or exploding, and network degradation. This resulted in the worse performance of the deep neural network [89]. He et al. [81] presented ResNet to address these issues. The basic architecture of ResNet [81] was composed of residual mapping $F(x)$ and identity mapping x , as shown in Figure 2B. The identity mapping ensured no loss of inputted information in spite of increasing layers. The residual mapping was viewed as the learnable residual function and might be conventional convolutions. The ResNet enabled the neural network to go deeper without network degradation. Li et al. [81] used ResNet to construct a 152-layers deep network, which reduced the top 5 error rates of image recognition to 5.71% on the ImageNet.

3.4. Bi-LSTM

Long-short term memory (LSTM) [90] is a type of recurrent neural network (RNN) [91,92]. The RNN is especially suitable to deal with time series questions due to its architecture: sharing weights at all of the time steps. The RNN was applied to a wide range of fields, including speech recognition [93], continuous B-cell epitope prediction [94], sentiment analysis [95], and action recognition [96]. The major default of the RNN was that it is prone to cause gradient vanishing or exploding for long sequence analysis. Therefore, the RNN was restricted to short sequences [97,98]. The LSTM [90] employed the gate mechanism to control conveying of information, including selective addition of new information or removal of information accumulated previously. The LSTM was able to capture the relationship of the words in the former with those in the back but was not able to characterize the relationship of the words in the back with those in the former. The Bi-LSTM [84,85] addressed the issue well. As shown in Figure 3, the Bi-LSTM was made up of two LSTMs, one from forward to backward and another from backward to forward. The two LSTMs shared embedding of words but were independent of each other in terms of learnable parameters. The concatenation of hidden states in both LSTMs corresponded to the output of the Bi-LSTM.

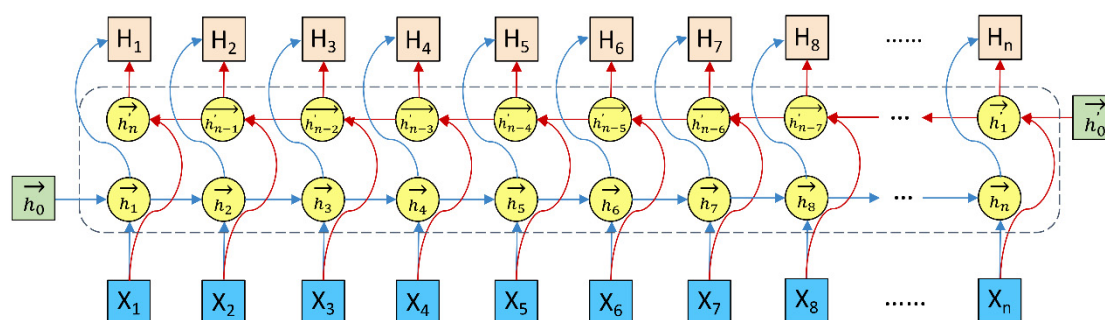


Figure 3. The diagram of the Bi-LSTM unit. X_i , \vec{h}_i , \overleftarrow{h}_{n-i} , and H_i denoted the input, the forward hidden state, the backward hidden state, and the bi-directional hidden state at the time step i , respectively.

3.5. Feed-Forward Attention

Attention mechanisms are increasingly becoming a hot topic in the field of deep learning. The attention mechanisms are a scheme of allocating weights, which is very similar to the scene where one assigns a different focus to different parts when watching an object. There are many attention schemes, including feed-forward attention [99] and self-attention [100], etc. The feed-forward attention is intended to make up for the deficiency of the LSTM in the long-term dependency. Assume that the hidden state at time step t in the LSTM was h_t . The context vector generated by the feed-forward attention was computed by

$$c = \sum_{t=1}^T \alpha_t h_t, \quad (2)$$

where α_t was the attention weight of the hidden state h_t . α_t was defined by

$$\alpha_t = \frac{\exp(e_t)}{\sum_{k=1}^T \exp(e_k)}, \quad (3)$$

where

$$e_t = \delta(h_t). \quad (4)$$

δ was the learnable parameter.

3.6. Dropout Layer

Dropout proposed by Hinton et al. [101] is a concept to train deep neural network. In the process of training, a certain proportion of neurons are randomly dropped out, and all of the neurons are used as usual in the process of prediction [102]. The dropout serves two-fold functions: speeding up training of the deep neural network and reducing over-fitting.

3.7. Flatten Layer and Fully Connected Layer

The flatten layer was intended to convert the shape of data so as to link conveniently the next layers. The flatten layers did not have any learnable parameters. The fully connected layer was identical to the hidden layer in the multilayer perceptron, and each neuron was connected to all of the neurons in the previous layer.

4. Cross Validation and Evaluation Metrics

To examine the predictive performance of the presented method, we used n -fold cross validation and independent test. In the n -fold cross-validation, the training dataset was divided into n parts of equal or approximately equal size, of which $n-1$ parts were used to train the model and the remaining part was used to test the model. This process was repeated n times. In the independent test, the training dataset was used to train the model, and the independent dataset was used to test the model.

This is a binary classification issue, so we used common metrics to evaluate the predictive performance, including sensitivity (SN), specificity (SP), accuracy (ACC), and Matthews' correlation coefficient (MCC), which are defined as

$$SN = \frac{TP}{TP + FN} \quad (5)$$

$$SP = \frac{TN}{FP + TN} \quad (6)$$

$$ACC = \frac{TP + TN}{TP + FN + FP + TN} \quad (7)$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FN)(TP + FP)(TN + FN)(TN + FP)}} \quad (8)$$

where TP is the number of true positive samples, FN is the number of false negative samples, FP is the number of false positive samples, and TN is the number of true negative samples. SN, SP and ACC lie between 0 and 1. The MCC ranges from -1 to 1 . More values of SN, SP, ACC and MCC indicated better performance.

The receiver operating characteristic (ROC) curve is a commonly used way to evaluate the performance of binary classification methods. The ROC curve is drawn by plotting the true positive rate (TPR) against the false positive rate (FPR) under various thresholds. The TPR and the FPR are computed by

$$TPR = \frac{TP}{TP + FN} \quad (9)$$

$$FPR = \frac{FP}{FP + TN} \quad (10)$$

The area under the ROC curve (AUC) ranges from 0 to 1. If the AUC was equal to 1, the prediction was perfect. The AUC equaling 0.5 indicated a random prediction and the AUC equaling to 0 was an opposite prediction.

We used Python programming language along with the deep learning toolkit TensorFlow (version 2.0) to implement the Enhancer-LSTMAtt. We conducted 5-fold cross validation, 10-fold cross validation and independent test on the Microsoft Windows 10 operating system, which is installed on a notebook computer with 32G RAM and 6 CPUs, each with 2.60 GHz. Each epoch costs about 25 s in the training process, while prediction of each sample takes no more than 2 s by using the trained Enhancer-LSTMAtt. The codes along with the datasets are available at Github: <https://github.com/feng-123/Enhancer-LSTMAtt>.

5. Results

We tested the Enhancer-LSTMAtt for its ability to not only distinguish between enhancers and non-enhancers, but also discriminate strong enhancers from weak enhancers. The process of distinguishing between enhancers and non-enhancers was called the first stage, where all of the enhancers, including weak enhancers, were positive samples. The process of discriminating strong from weak enhancers was called the second stage, where the strong enhancers were positive and the weak enhancers were negative samples. We conducted 5-fold cross validation in dataset S. Figure 4 shows the ROC curve of each fold, and Table 2 lists the evaluation of performance. We obtained an average AUC of 0.8259 in the first stage and an average AUC of 0.6439 in the second stage. We achieved an average SN of 0.7304, an average SP of 0.8006, an average ACC of 0.7655, and an average MCC of 0.5339 in the first stage and an average SN of 0.6765, an average SP of 0.6024, an average ACC of 0.6395, and an average MCC of 0.2804 in the second stage. Obviously, the predictive performance in the first stage was much better than that in the second stage, indicating that it was more difficult to discriminate strong enhancers from weak enhancers than to discriminate enhancers from non-enhancers.

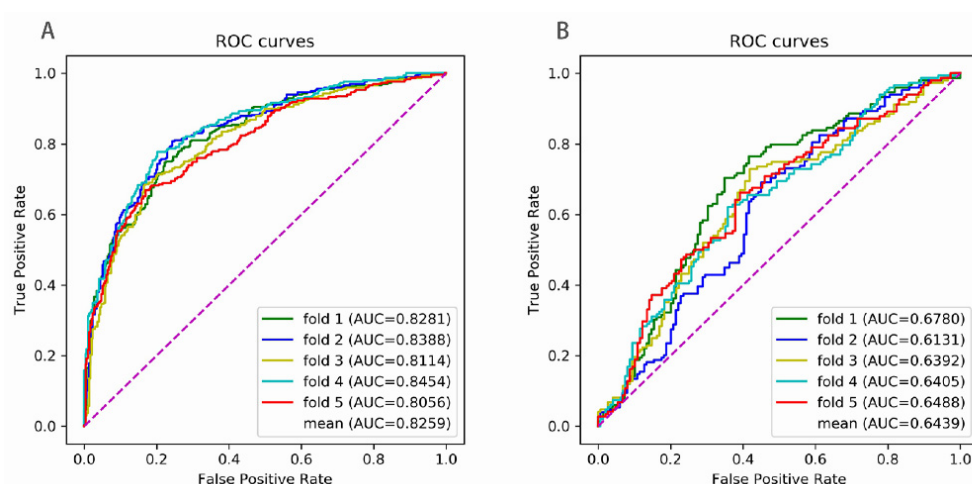


Figure 4. The ROC curves of 5-fold cross validation (A) for recognizing the enhancers and (B) for distinguishing strong from weak enhancers. The purple line is the baseline of ROC curve, named random guessing line.

Table 2. Performances of 5-fold cross validation.

	SN	SP	ACC	MCC	AUC
Frist Stage					
fold 1	0.7374	0.7811	0.7593	0.5190	0.8281
fold 2	0.8013	0.7576	0.7795	0.5595	0.8388
fold 3	0.6801	0.8350	0.7576	0.5214	0.8114
fold 4	0.7710	0.7980	0.7845	0.5692	0.8454
fold 5	0.6622	0.8311	0.7466	0.5004	0.8056
mean	0.7304	0.8006	0.7655	0.5339	0.8259
Second Stage					
fold 1	0.6846	0.6510	0.6678	0.3358	0.6780
fold 2	0.6913	0.5369	0.6141	0.2310	0.6131
fold 3	0.7297	0.5811	0.6554	0.3143	0.6392
fold 4	0.6149	0.6419	0.6284	0.2569	0.6405
fold 5	0.6622	0.6014	0.6318	0.2640	0.6488
mean	0.6765	0.6024	0.6395	0.2804	0.6439

5.1. Comparison with State-of-the-Art Methods

As mentioned in the introduction, no less than 20 computational methods have been developed for predicting enhancers. Some methods were tested by jackknife test, some by 5-fold cross validation, some by 10-fold cross-validation, and some by the independent test. Some methods distinguished enhancers from non-enhancers, while some discriminated strong from weak enhancers. Table 3 summarizes these methods. Since the jackknife test is too time-consuming for deep learning methods, we conducted 5-fold cross, 10-fold cross validation, and independent tests to compare these state-of-the-art methods. Tables 4 and 5 list the evaluation of performances. Different indices evaluate different performances. For instance, SN is used to evaluate the ratio of the number of correctly predicted positive samples to the total number of positive ones, while SP is the ratio of the number of correctly predicted negative samples to the total number of negative ones. Sometimes, the two indices would not maintain synchronization, which was difficult to determine as good or bad. In this case, the overall indices could be used, such as ACC and MCC. In the 5-fold cross-validation, Enhancer-LSTMAtt was superior to Enhancer-BERT [55], DeployEnhancer [48] and iEnhancer-RF [57] in terms of ACC and MCC in the first stage and exceeded iEnhancer-PsedeKNC [41], DeployEnhancer [48], EnhancerP-2L [51], and iEnhancer-RF [57] in terms of MCC in the second stage. In the 10-fold cross-validation,

Enhancer-LSTMAtt reached competitive performance with ES-ARCNN [49], iEnhancer-XG [53], and iEnhancer-MFGBDT [63] in the second stage.

Table 3. Summary of the state-of-the-art methods for predicting enhancers.

Method	Jackknife	5-Fold	10-Fold	Independent	Enhancer or Not	Strong or Weak
iEnhancer-2L [40]	✓			✓	✓	✓
iEnhancer-PsedeKNC [41]		✓			✓	✓
EnhancerPred [42]	✓			✓	✓	✓
EnhancerPred2.0 [43]	✓				✓	✓
Enhancer-Tri-N [44]	✓				✓	✓
iEnhancer-2L-Hybrid [45]	✓				✓	✓
iEnhancer-EL [46]	✓			✓	✓	✓
iEnhancer-5Step [47]		✓		✓	✓	✓
DeployEnhancer [48]		✓		✓	✓	✓
ES-ARCNN [49]			✓	✓	✓	✓
iEnhancer-ECNN [50]				✓	✓	✓
EnhancerP-2L [51]		✓	✓	✓	✓	✓
iEnhancer-CNN [52]		✓		✓	✓	✓
iEnhancer-XG [53]			✓	✓	✓	✓
Enhancer-DRRNN [54]				✓	✓	✓
Enhancer-BERT [55]		✓		✓	✓	✓
iEnhancer-KL [56]		✓		✓	✓	✓
iEnhancer-RF [57]		✓		✓	✓	✓
spEnhancer [58]				✓	✓	✓
iEnhancer-EBLSTM [59]				✓	✓	✓
iEnhancer-GAN [60]			✓	✓	✓	✓
piEnPred [61]		✓		✓	✓	✓
iEnhancer-RD [62]		✓		✓	✓	✓
iEnhancer-MFGBDT [63]			✓	✓	✓	✓

✓ denoted conduction of corresponding cross validation.

Table 4. Comparison with state-of-the-art methods by 5-fold cross validation.

	SN	SP	ACC	MCC	AUC
Frist Stage					
iEnhancer-PsedeKNC [41]	0.7731	0.7630	0.7678	0.5400	0.8500
iEnhancer-5Step [47]	0.8110	0.8350	0.8230	0.6500	-
DeployEnhancer [48]	0.7325	0.7642	0.7483	0.4980	0.7694
EnhancerP-2L [51]	0.9077	0.9259	0.9168	0.8340	0.9400
iEnhancer-CNN [52]	0.7588	0.8888	0.8063	0.6929	0.8957
Enhancer-BERT [55]	0.7950	0.7300	0.7620	0.5250	-
iEnhancer-KL [56]	0.8322	0.8524	0.8423	0.6800	-
iEnhancer-RF [57]	0.7364	0.7871	0.7618	0.5264	0.8400
piEnPred [61]	0.9228	0.8047	0.8788	0.7660	0.9603
iEnhancer-RD [62]	0.8100	0.7650	0.7880	0.5760	0.8440
Enhancer-LSTMAtt	0.7304	0.8006	0.7655	0.5339	0.8259
Second Stage					
iEnhancer-PsedeKNC [41]	0.6262	0.6441	0.6341	0.2700	0.6900
iEnhancer-5Step [47]	0.7530	0.6080	0.6810	0.3700	-
DeployEnhancer [48]	0.7965	0.3828	0.5896	0.1970	0.6068
EnhancerP-2L [51]	0.6221	0.6182	0.6193	0.2400	0.9000
iEnhancer-CNN [52]	0.7364	0.7680	0.7643	0.4505	0.8109
iEnhancer-KL [56]	0.9340	0.9287	0.9313	0.8600	-
iEnhancer-RF [57]	0.6846	0.5661	0.6253	0.2529	0.6700
piEnPred [61]	0.6554	0.7094	68.24	0.3654	0.7568
iEnhancer-RD [62]	0.8400	0.5700	0.7050	0.4260	0.7920
Enhancer-LSTMAtt	0.6765	0.6024	0.6395	0.2804	0.6439

Table 5. Comparison with state-of-the-art methods by 10-fold cross validation.

	SN	SP	ACC	MCC	AUC
Frist Stage					
EnhancerP-2L [51]	0.8653	0.9690	0.9172	0.8398	0.9700
iEnhancer-XG [53]	0.7570	0.8650	0.8110	0.6265	-
iEnhancer-GAN [60]	0.9510	0.9510	0.9510	0.9020	-
iEnhancer-MFGBDT [63]	0.7754	0.7978	0.7867	0.5735	-
Enhancer-LSTMAtt	0.7414	0.7873	0.7658	0.5298	0.8256
Second Stage					
ES-ARCNN [49]	0.7278	59.57	66.17	0.3263	0.6604
EnhancerP-2L [51]	0.8049	0.9397	0.8723	0.7519	0.9300
iEnhancer-XG [53]	0.7494	0.5855	0.6674	0.3395	-
iEnhancer-GAN [60]	0.8730	0.8710	0.8720	0.7440	-
iEnhancer-MFGBDT [63]	0.7056	0.6163	0.6604	0.3232	-
Enhancer-LSTMAtt	0.6463	0.6380	0.6429	0.2851	0.6550

Table 6 lists evaluation of performances of all of the 19 methods on the independent test. To the best of our knowledge, nearly all of the methods used the same independent dataset S^i for independent test, and no other published enhancers were collected as the second independent dataset. Obviously, Enhancer-LSTMAtt achieved competitive performance with these state-of-the-art methods. In the first stage, Enhancer-LSTMAtt reached the best SP (0.8150), the best ACC (0.8050), and the best MCC (0.6101), achieved a second AUC (0.8588), which was less than the AUC of iEnhancer-RF, and obtained a competitive SN (0.7950), which was less than the SN of iEnhancer-GAN [60], spEnhancer [58], iEnhancer-5Step [47], piEnPred [61], iEnhancer-RD [62], and iEnhancer-BERT [55]. In the second stage, the Enhancer-LSTMAtt reached the best SN, ACC and MCC, a second AUC to that of the iEnhancer-RF [57], and a second SP to that of the Enhancer-DRRNN [54]. These results indicated that Enhancer-LSTMAtt is a competitive method to recognize enhancers. It must be pointed out that we didn't conduct cross validation and independent test for 19 methods, and the evaluation of their performances directly came from their published papers. Figure 5 shows the ROC curve of the independent test.

Table 6. Comparison with state-of-the-art methods by independent test.

	SN	SP	ACC	MCC	AUC
Frist Stage					
iEnhancer-2L [40]	0.7100	0.7500	0.7300	0.4604	0.8062
EnhancerPred [42]	0.7350	0.7450	0.7400	0.4800	0.8013
iEnhancer-EL [46]	0.7100	0.7850	0.7475	0.4964	0.8173
iEnhancer-5Step [47]	0.8200	0.7600	0.7900	0.5800	-
DeployEnhancer [48]	0.7550	0.7600	0.7550	0.5100	0.7704
iEnhancer-ECNN [50]	0.7520	0.7850	0.7690	0.5370	0.8320
EnhancerP-2L [51]	0.7810	0.8105	0.7950	0.5907	-
iEnhancer-CNN [52]	0.7825	0.7900	0.7750	0.5850	-
iEnhancer-XG [53]	0.7400	0.7750	0.7575	0.5150	-
Enhancer-DRRNN [54]	0.7330	0.8010	0.7670	0.5350	0.8370
Enhancer-BERT [55]	0.8000	0.7120	0.7560	0.5140	-
iEnhancer-RF [57]	0.7850	0.8100	0.7975	0.5952	0.8600
spEnhancer [58]	0.8300	0.7150	0.7725	0.5793	0.8235
iEnhancer-EBLSTM [59]	0.7550	0.7950	0.7720	0.5340	0.8350
iEnhancer-GAN [60]	0.8110	0.7580	0.7840	0.5670	-
piEnPred [61]	0.8250	0.7840	0.8040	0.6099	-
iEnhancer-RD [62]	0.8100	0.7650	0.7880	0.5760	0.8440
iEnhancer-MFGBDT [63]	0.7679	0.7955	0.7750	0.5607	-
Enhancer-LSTMAtt	0.7950	0.8150	0.8050	0.6101	0.8588

Table 6. Cont.

	SN	SP	ACC	MCC	AUC
Second Stage					
iEnhancer-2L [40]	0.4700	0.7400	0.6050	0.2181	0.6678
EnhancerPred [42]	0.4500	0.6500	0.5500	0.1020	0.5790
iEnhancer-EL [46]	0.5400	0.6800	0.6100	0.2222	0.6801
iEnhancer-5Step [47]	0.7400	0.5300	0.6350	0.2800	-
DeployEnhancer [48]	0.8315	0.4561	0.6849	0.3120	0.6714
ES-ARCNN [49]	0.8600	0.4500	0.6560	0.3399	-
iEnhancer-ECNN [50]	0.7910	0.5640	0.6780	0.3680	0.7480
EnhancerP-2L [51]	0.6829	0.7922	0.7250	0.4624	-
iEnhancer-CNN [52]	0.6525	0.7610	0.7500	0.3232	-
iEnhancer-XG [53]	0.7000	0.5700	0.6350	0.2720	-
Enhancer-DRRNN [54]	0.8580	0.8400	0.8490	0.6990	-
iEnhancer-RF [57]	0.9300	0.7700	0.8500	0.7091	0.9700
spEnhancer [58]	0.9100	0.3300	0.6200	0.3703	0.6253
iEnhancer-EBLSTM [59]	0.8120	0.5360	0.6580	0.3240	0.6880
iEnhancer-GAN [60]	0.9610	0.5370	0.7490	0.5050	-
piEnPred [61]	0.7000	0.7500	0.7250	0.4506	-
iEnhancer-RD [62]	0.8400	0.5700	0.7050	0.4260	0.7920
iEnhancer-MFGBDT [63]	0.7255	0.6681	0.6850	0.3862	-
Enhancer-LSTMAtt	0.9900	0.8000	0.8950	0.8047	0.9637

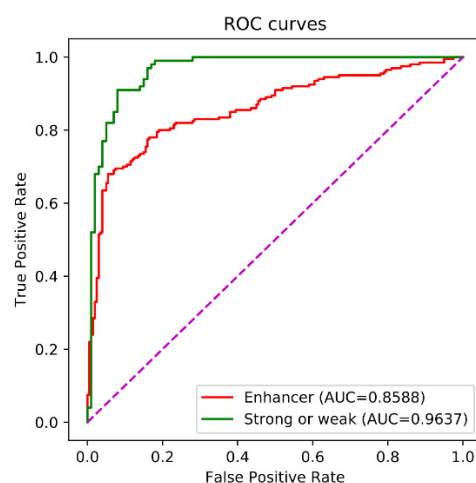


Figure 5. The ROC curves of the independent test. The purple line is the baseline of ROC curve, named random guess line.

5.2. Enhancer-LSTMAtt Webservice

We implemented Enhancer-LSTMAtt into a user-friendly web application which is freely available at <http://www.biolscience.cn/Enhancer-LSTMAtt/> (accessed on 20 May 2022) to all of the scientific researchers. The web application is easy for users to use. The only thing users do is to upload DNA sequences in a FASTA format either by pasting it into textbox or by uploading a file. Users click the “submit” button, and then the web application returns prediction in the 7-tuple. The first column is the names of input sequences, the second is the range of the enhancers, the third and the fourth are the probabilities of predicting as the enhancer and the non-enhancer respectively, the fifth and the sixth are the probabilities of predicting as the strong and the weak enhancer, and the seventh is the predicted result.

5.3. Discussion

We investigated effect of different non-enhancers on the methods. Due to non-enhancers that were not available before sampling, we used the sampling and mutation

strategy to generate new non-enhancers. We randomly selected 30%, 40%, and 50% of samples in the non-enhancer set S_{non} and made them mutate. The mutated non-enhancers and the non-mutated non-enhancers constituted three new non-enhancer sets which along with the enhancers further comprised three new training sets, respectively. We used the independent test to examine the performance of the proposed method trained by the new training sets. As shown in Figure 6, the non-enhancers have a certain influence on the performance of the method, but this influence is little.

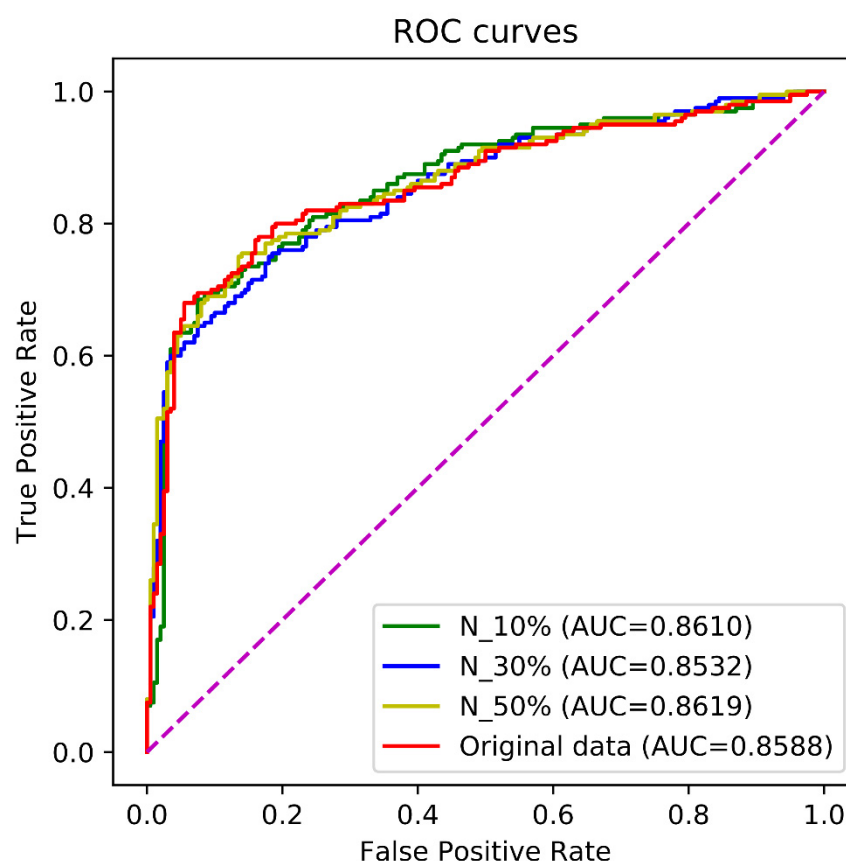


Figure 6. The ROC curves by the independent test over different non-enhancer sets. N_10, N_30, and N_50 denote the training sets in which 10%, 30%, 50% of non-enhancers were formed by mutation from the original non-enhancers, respectively. The original data denotes the training set which was made up of the enhancers and original non-enhancers. The purple line is the baseline of ROC curve, named random guessing line.

In the Enhancer-LSTMAtt, there are up to 250,921 trainable parameters. The more trainable parameters there are, the more overfitting the deep learning model. We used dropout and batch normalization to reduce model overfitting. We investigated the roles of both techniques in reducing overfitting. As shown in Figure 7, the training loss descended rapidly at the beginning stage and then slowly declined to be stable with the increment of epoch, while the loss of the independent test declined rapidly at the beginning stage and then fluctuated in a certain range. The AUC of the independent test ascended rapidly at the beginning stage and then tended to stabilize with the increment of epochs. Therefore, there is no remarkable overfitting issues for Enhancer-LSTMAtt.

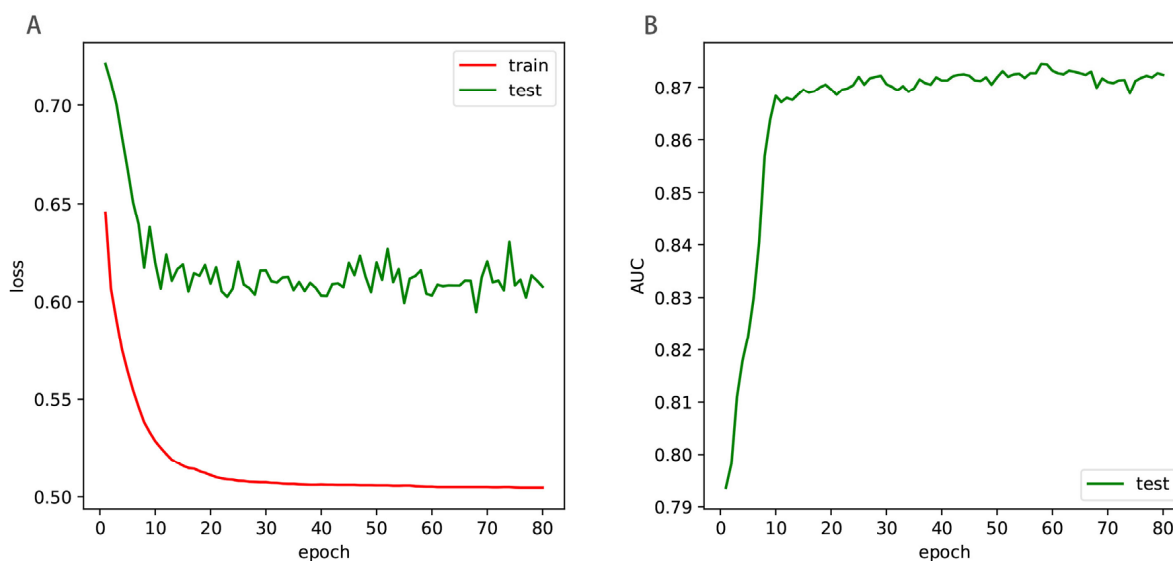


Figure 7. The loss curves (A) and the AUC curve of the independent test (B). The ‘train’ denoted the training loss in the benchmark dataset, and the ‘test’ denoted the loss in the training set and AUC values of the independent test.

Enhancers-LSTMAtt is a deep learning-based and end-to-end method that does not require any feature design. This avoided artificial interference and sophisticated feature extraction or selection. The Enhancers-LSTMAtt is easier to implement than the feature-based methods from this viewpoint. Most feature-based methods performed well over the cross validation but performed badly over the independent test, indicating the weakly generalized ability. For example, the EnhancerP-2L [51] achieved an MCC of 0.8340 and an MCC of 0.8398 over the 5-fold and 10-fold cross validations, respectively, but reached only an MCC of 0.5907, which decreased by more than 0.24. piEnPred [61] substantially decreased the MCC from 0.7660 over the 5-fold cross validation to 0.6099 over the independent test. The Enhancers-LSTMAtt did not reduce the MCC over the independent test and instead increased the MCC by at least 0.07. Thus, the Enhancers-LSTMAtt is more generalized to the independent test than the feature-based methods. Most deep learning-based methods either utilize the CNN, LSTM, or their combination for enhancer recognition. For example, both the iEnhancer-ECNN [50] and the iEnhancer-CNN [52] exploited the CNN, the iEnhancer-EBLSTM [59] used Bi-LSTM, and the DeployEnhancer [48] sequentially combined the CNN and Bi-LSTM. The CNN and Bi-LSTM are two popular neural network architectures that have the ability to capture different information. The sequential combination between CNN and Bi-LSTM is disadvantageous to complete exploitation of these two different types of information. Stacking the CNN and Bi-LSTM in a parallel manner is able to exploit their respective representations. In addition, we also used the residual network and the attention mechanism to improve representation. This is two potential reasons why Enhancers-LSTMAtt is superior to other deep learning-based methods in the independent test, such as the iEnhancer-ECNN [50], the iEnhancer-CNN [52], the iEnhancer-EBLSTM [59], and the DeployEnhancer [48]. On the other hand, inclusion of the residual neural network as well as the feed-forward attention and stacking CNN and Bi-LSTM in a parallel manner added complexity to a certain extent, which in turn increased the computing cost.

6. Conclusions

Identifying enhancers is key to uncovering their roles in the regulation of transcription for target genes. We employed multiple deep learning techniques (i.e., Bi-LSTM, CNN, residual network and feed-forward attention) to construct Enhancer-LSTMAtt for enhancer recognition. The Enhancer-LSTMAtt is of the following superiorities over the

state-of-the-art methods: (1) the Enhancer-LSTMAtt stacked the CNN and the LSTM in parallel, not in a series-connection manner, which allows stacking diverse representations; (2) the Enhancer-LSTMAtt utilized the residual neural network, which allows construction of deeper neural networks without loss of information; and (3) the Enhancer-LSTMAtt employed the attention mechanism, which allows focusing on key information. Comprehensive comparison with state-of-the-art methods suggested that Enhancers-LSTMAtt was not only a stable tool but also an effective and efficient tool for enhancer identification.

Author Contributions: Conceptualization, G.H., Y.Y., Y.L. and D.-Q.W.; methodology, G.H., W.L. and G.Z.; software, G.Z. and P.Z.; validation, G.Z.; formal analysis, W.L.; investigation, W.L. and G.Z.; resources, G.H.; data curation, G.Z.; writing—original draft preparation, G.H., W.L., G.Z., Y.Y. and J.L.; writing—review and editing, G.H., W.L. and G.Z.; visualization, G.Z.; supervision, G.H.; project administration, G.H.; funding acquisition, G.H., Y.Y. and Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China [62162025, 61672356], by Hunan Provincial Natural Science Foundation of China [2022JJ50177, 2020JJ4034], by Scientific Research Fund of Hunan Provincial Education Department [21A0466, 19A215], and by Shaoyang University Innovation Foundation for Postgraduate [CX2021SY033].

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: <https://github.com/feng-123/Enhancer-LSTMAtt>.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Blackwood, E.M.; Kadonaga, J.T. Going the distance: A current view of enhancer action. *Science* **1998**, *281*, 60–63. [[CrossRef](#)] [[PubMed](#)]
2. Pennacchio, L.A.; Bickmore, W.; Dean, A.; Nobrega, M.A.; Bejerano, G. Enhancers: Five essential questions. *Nat. Rev. Genet.* **2013**, *14*, 288–295. [[CrossRef](#)] [[PubMed](#)]
3. Maston, G.A.; Evans, S.K.; Green, M.R. Transcriptional regulatory elements in the human genome. *Annu. Rev. Genom. Hum. Genet.* **2006**, *7*, 29–59. [[CrossRef](#)] [[PubMed](#)]
4. Grosveld, F.; van Staalduinen, J.; Stadhouders, R. Transcriptional Regulation by (Super) Enhancers: From Discovery to Mechanisms. *Annu. Rev. Genom. Hum. Genet.* **2021**, *22*, 127–146. [[CrossRef](#)] [[PubMed](#)]
5. Shlyueva, D.; Stampfel, G.; Stark, A. Transcriptional enhancers: From properties to genome-wide predictions. *Nat. Rev. Genet.* **2014**, *15*, 272–286. [[CrossRef](#)] [[PubMed](#)]
6. Parker, S.C.; Stitzel, M.L.; Taylor, D.L.; Orozco, J.M.; Erdos, M.R.; Akiyama, J.A.; van Bueren, K.L.; Chines, P.S.; Narisu, N.; Black, B.L. Chromatin stretch enhancer states drive cell-specific gene regulation and harbor human disease risk variants. *Proc. Natl. Acad. Sci. USA* **2013**, *110*, 17921–17926. [[CrossRef](#)]
7. Schoenfelder, S.; Fraser, P. Long-range enhancer–promoter contacts in gene expression control. *Nat. Rev. Genet.* **2019**, *20*, 437–455. [[CrossRef](#)]
8. Chan, Y.F.; Marks, M.E.; Jones, F.C.; Villarreal, G.; Shapiro, M.D.; Brady, S.D.; Southwick, A.M.; Absher, D.M.; Grimwood, J.; Schmutz, J. Adaptive evolution of pelvic reduction in sticklebacks by recurrent deletion of a Pitx1 enhancer. *Science* **2010**, *327*, 302–305. [[CrossRef](#)]
9. Levine, M. Transcriptional enhancers in animal development and evolution. *Curr. Biol.* **2010**, *20*, R754–R763. [[CrossRef](#)]
10. Bonn, S.; Zinzen, R.P.; Girardot, C.; Gustafson, E.H.; Perez-Gonzalez, A.; Delhomme, N.; Ghavi-Helm, Y.; Wilczyński, B.; Riddell, A.; Furlong, E.E. Tissue-specific analysis of chromatin state identifies temporal signatures of enhancer activity during embryonic development. *Nat. Genet.* **2012**, *44*, 148–156. [[CrossRef](#)]
11. Heintzman, N.D.; Hon, G.C.; Hawkins, R.D.; Kheradpour, P.; Stark, A.; Harp, L.F.; Ye, Z.; Lee, L.K.; Stuart, R.K.; Ching, C.W. Histone modifications at human enhancers reflect global cell-type-specific gene expression. *Nature* **2009**, *459*, 108–112. [[CrossRef](#)] [[PubMed](#)]
12. Visel, A.; Blow, M.J.; Li, Z.; Zhang, T.; Akiyama, J.A.; Holt, A.; Plajzer-Frick, I.; Shoukry, M.; Wright, C.; Chen, F. ChIP-seq accurately predicts tissue-specific activity of enhancers. *Nature* **2009**, *457*, 854–858. [[CrossRef](#)]
13. Heintzman, N.D.; Stuart, R.K.; Hon, G.; Fu, Y.; Ching, C.W.; Hawkins, R.D.; Barrera, L.O.; Van Calcar, S.; Qu, C.; Ching, K.A. Distinct and predictive chromatin signatures of transcriptional promoters and enhancers in the human genome. *Nat. Genet.* **2007**, *39*, 311–318. [[CrossRef](#)]
14. Jin, F.; Li, Y.; Ren, B.; Natarajan, R. PU. 1 and C/EBP α synergistically program distinct response to NF- κ B activation through establishing monocyte specific enhancers. *Proc. Natl. Acad. Sci. USA* **2011**, *108*, 5290–5295. [[CrossRef](#)] [[PubMed](#)]

15. Kim, T.-K.; Hemberg, M.; Gray, J.M.; Costa, A.M.; Bear, D.M.; Wu, J.; Harmin, D.A.; Laptewicz, M.; Barbara-Haley, K.; Kuersten, S. Widespread transcription at neuronal activity-regulated enhancers. *Nature* **2010**, *465*, 182–187. [[CrossRef](#)] [[PubMed](#)]
16. Rajagopal, N.; Xie, W.; Li, Y.; Wagner, U.; Wang, W.; Stamatoyannopoulos, J.; Ernst, J.; Kellis, M.; Ren, B. RFECFS: A random-forest based algorithm for enhancer identification from chromatin state. *PLoS Comput. Biol.* **2013**, *9*, e1002968. [[CrossRef](#)]
17. Whyte, W.A.; Orlando, D.A.; Hnisz, D.; Abraham, B.J.; Lin, C.Y.; Kagey, M.H.; Rahl, P.B.; Lee, T.I.; Young, R.A. Master transcription factors and mediator establish super-enhancers at key cell identity genes. *Cell* **2013**, *153*, 307–319. [[CrossRef](#)]
18. Kleftogiannis, D.; Kalnis, P.; Bajic, V.B. Progress and challenges in bioinformatics approaches for enhancer identification. *Brief. Bioinform.* **2016**, *17*, 967–979. [[CrossRef](#)]
19. Johnson, D.S.; Mortazavi, A.; Myers, R.M.; Wold, B. Genome-wide mapping of in vivo protein-DNA interactions. *Science* **2007**, *316*, 1497–1502. [[CrossRef](#)]
20. Robertson, G.; Hirst, M.; Bainbridge, M.; Bilenky, M.; Zhao, Y.; Zeng, T.; Euskirchen, G.; Bernier, B.; Varhol, R.; Delaney, A. Genome-wide profiles of STAT1 DNA association using chromatin immunoprecipitation and massively parallel sequencing. *Nat. Methods* **2007**, *4*, 651–657. [[CrossRef](#)]
21. Bulyk, M.L.; Gentalen, E.; Lockhart, D.J.; Church, G.M. Quantifying DNA–protein interactions by double-stranded DNA arrays. *Nat. Biotechnol.* **1999**, *17*, 573–577. [[CrossRef](#)] [[PubMed](#)]
22. Tuerk, C.; Gold, L. Systematic evolution of ligands by exponential enrichment: RNA ligands to bacteriophage T4 DNA polymerase. *Science* **1990**, *249*, 505–510. [[CrossRef](#)]
23. Li, J.J.; Herskowitz, I. Isolation of ORC6, a component of the yeast origin recognition complex by a one-hybrid system. *Science* **1993**, *262*, 1870–1874. [[CrossRef](#)] [[PubMed](#)]
24. Meng, X.; Brodsky, M.H.; Wolfe, S.A. A bacterial one-hybrid system for determining the DNA-binding specificity of transcription factors. *Nat. Biotechnol.* **2005**, *23*, 988–994. [[CrossRef](#)] [[PubMed](#)]
25. Heintzman, N.D.; Ren, B. Finding distal regulatory elements in the human genome. *Curr. Opin. Genet. Dev.* **2009**, *19*, 541–549. [[CrossRef](#)]
26. May, D.; Blow, M.J.; Kaplan, T.; McCulley, D.J.; Jensen, B.C.; Akiyama, J.A.; Holt, A.; Plajzer-Frick, I.; Shoukry, M.; Wright, C. Large-scale discovery of enhancers from human heart tissue. *Nat. Genet.* **2012**, *44*, 89–93. [[CrossRef](#)] [[PubMed](#)]
27. Boyle, A.P.; Song, L.; Lee, B.-K.; London, D.; Keefe, D.; Birney, E.; Iyer, V.R.; Crawford, G.E.; Furey, T.S. High-resolution genome-wide in vivo footprinting of diverse transcription factors in human cells. *Genome Res.* **2011**, *21*, 456–464. [[CrossRef](#)] [[PubMed](#)]
28. Boyle, A.P.; Davis, S.; Shulha, H.P.; Meltzer, P.; Margulies, E.H.; Weng, Z.; Furey, T.S.; Crawford, G.E. High-resolution mapping and characterization of open chromatin across the genome. *Cell* **2008**, *132*, 311–322. [[CrossRef](#)]
29. Consortium, E.P. Identification and analysis of functional elements in 1% of the human genome by the ENCODE pilot project. *Nature* **2007**, *447*, 799. [[CrossRef](#)]
30. Visel, A.; Bristow, J.; Pennacchio, L.A. Enhancer identification through comparative genomics. *Proc. Semin. Cell Dev. Biol.* **2007**, *18*, 140–152. [[CrossRef](#)]
31. Won, K.-J.; Zhang, X.; Wang, T.; Ding, B.; Raha, D.; Snyder, M.; Ren, B.; Wang, W. Comparative annotation of functional regions in the human genome using epigenomic data. *Nucleic Acids Res.* **2013**, *41*, 4423–4432. [[CrossRef](#)] [[PubMed](#)]
32. Ghandi, M.; Lee, D.; Mohammad-Noori, M.; Beer, M.A. Enhanced regulatory sequence prediction using gapped k-mer features. *PLoS Comput. Biol.* **2014**, *10*, e1003711. [[CrossRef](#)] [[PubMed](#)]
33. Min, X.; Zeng, W.; Chen, S.; Chen, N.; Chen, T.; Jiang, R. Predicting enhancers with deep convolutional neural networks. *BMC Bioinform.* **2017**, *18*, 35–46. [[CrossRef](#)]
34. Yang, B.; Liu, F.; Ren, C.; Ouyang, Z.; Xie, Z.; Bo, X.; Shu, W. BiRen: Predicting enhancers with a deep-learning-based model using the DNA sequence alone. *Bioinformatics* **2017**, *33*, 1930–1936. [[CrossRef](#)]
35. Firpi, H.A.; Ucar, D.; Tan, K. Discover regulatory DNA elements using chromatin signatures and artificial neural network. *Bioinformatics* **2010**, *26*, 1579–1586. [[CrossRef](#)] [[PubMed](#)]
36. Fernandez, M.; Miranda-Saavedra, D. Genome-wide enhancer prediction from epigenetic signatures using genetic algorithm-optimized support vector machines. *Nucleic Acids Res.* **2012**, *40*, e77. [[CrossRef](#)]
37. Erwin, G.D.; Oksenberg, N.; Truty, R.M.; Kostka, D.; Murphy, K.K.; Ahituv, N.; Pollard, K.S.; Capra, J.A. Integrating diverse datasets improves developmental enhancer prediction. *PLoS Comput. Biol.* **2014**, *10*, e1003677. [[CrossRef](#)]
38. Kleftogiannis, D.; Kalnis, P.; Bajic, V.B. DEEP: A general computational framework for predicting enhancers. *Nucleic Acids Res.* **2015**, *43*, e6. [[CrossRef](#)]
39. Lu, Y.; Qu, W.; Shan, G.; Zhang, C. DELTA: A distal enhancer locating tool based on AdaBoost algorithm and shape features of chromatin modifications. *PLoS ONE* **2015**, *10*, e0130622. [[CrossRef](#)]
40. Liu, B.; Fang, L.; Long, R.; Lan, X.; Chou, K.-C. iEnhancer-2L: A two-layer predictor for identifying enhancers and their strength by pseudo k-tuple nucleotide composition. *Bioinformatics* **2016**, *32*, 362–369. [[CrossRef](#)]
41. Liu, B. iEnhancer-PsedeKNC: Identification of enhancers and their subgroups based on Pseudo degenerate kmer nucleotide composition. *Neurocomputing* **2016**, *217*, 46–52. [[CrossRef](#)]
42. Jia, C.; He, W. EnhancerPred: A predictor for discovering enhancers based on the combination and selection of multiple features. *Sci. Rep.* **2016**, *6*, 38741. [[CrossRef](#)]

43. He, W.; Jia, C. EnhancerPred2. 0: Predicting enhancers and their strength based on position-specific trinucleotide propensity and electron–ion interaction potential feature selection. *Mol. Biosyst.* **2017**, *13*, 767–774. [[CrossRef](#)] [[PubMed](#)]
44. Tahir, M.; Hayat, M.; Kabir, M. Sequence based predictor for discrimination of enhancer and their types by applying general form of Chou’s trinucleotide composition. *Comput. Methods Programs Biomed.* **2017**, *146*, 69–75. [[CrossRef](#)] [[PubMed](#)]
45. Tahir, M.; Hayat, M.; Khan, S.A. A two-layer computational model for discrimination of enhancer and their types using hybrid features pace of pseudo K-tuple nucleotide composition. *Arab. J. Sci. Eng.* **2018**, *43*, 6719–6727. [[CrossRef](#)]
46. Liu, B.; Li, K.; Huang, D.-S.; Chou, K.-C. iEnhancer-EL: Identifying enhancers and their strength with ensemble learning approach. *Bioinformatics* **2018**, *34*, 3835–3842. [[CrossRef](#)]
47. Le, N.Q.K.; Yapp, E.K.Y.; Ho, Q.-T.; Nagasundaram, N.; Ou, Y.-Y.; Yeh, H.-Y. iEnhancer-5Step: Identifying enhancers using hidden information of DNA sequences via Chou’s 5-step rule and word embedding. *Anal. Biochem.* **2019**, *571*, 53–61. [[CrossRef](#)]
48. Tan, K.K.; Le, N.Q.K.; Yeh, H.-Y.; Chua, M.C.H. Ensemble of deep recurrent neural networks for identifying enhancers via dinucleotide physicochemical properties. *Cells* **2019**, *8*, 767. [[CrossRef](#)]
49. Zhang, T.-H.; Flores, M.; Huang, Y. ES-ARCNN: Predicting enhancer strength by using data augmentation and residual convolutional neural network. *Anal. Biochem.* **2021**, *618*, 114120. [[CrossRef](#)]
50. Nguyen, Q.H.; Nguyen-Vo, T.-H.; Le, N.Q.K.; Do, T.T.; Rahardja, S.; Nguyen, B.P. iEnhancer-ECNN: Identifying enhancers and their strength using ensembles of convolutional neural networks. *BMC Genom.* **2019**, *20*, 951. [[CrossRef](#)]
51. Butt, A.H.; Alkhalaf, S.; Iqbal, S.; Khan, Y.D. EnhancerP-2L: A Gene regulatory site identification tool for DNA enhancer region using CREs motifs. *bioRxiv* **2020**. [[CrossRef](#)]
52. Khanal, J.; Tayara, H.; Chong, K.T. Identifying enhancers and their strength by the integration of word embedding and convolution neural network. *IEEE Access* **2020**, *8*, 58369–58376. [[CrossRef](#)]
53. Cai, L.; Ren, X.; Fu, X.; Peng, L.; Gao, M.; Zeng, X. iEnhancer-XG: Interpretable sequence-based enhancers and their strength predictor. *Bioinformatics* **2021**, *37*, 1060–1067. [[CrossRef](#)] [[PubMed](#)]
54. Li, Q.; Xu, L.; Li, Q.; Zhang, L. Identification and classification of enhancers using dimension reduction technique and recurrent neural network. *Comput. Math. Methods Med.* **2020**, *2020*, 8852258. [[CrossRef](#)] [[PubMed](#)]
55. Le, N.Q.K.; Ho, Q.-T.; Nguyen, T.-T.-D.; Ou, Y.-Y. A transformer architecture based on BERT and 2D convolutional neural network to identify DNA enhancers from sequence information. *Brief. Bioinform.* **2021**, *22*, bbab005. [[CrossRef](#)]
56. Lyu, Y.; Zhang, Z.; Li, J.; He, W.; Ding, Y.; Guo, F. iEnhancer-KL: A novel two-layer predictor for identifying enhancer by position specific of nucleotide composition. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2021**, *18*, 2809–2815. [[CrossRef](#)]
57. Lim, D.Y.; Khanal, J.; Tayara, H.; Chong, K.T. iEnhancer-RF: Identifying enhancers and their strength by enhanced feature representation using random forest. *Chemom. Intell. Lab. Syst.* **2021**, *212*, 104284. [[CrossRef](#)]
58. Mu, X.; Wang, Y.; Duan, M.; Liu, S.; Li, F.; Wang, X.; Zhang, K.; Huang, L.; Zhou, F. A Novel Position-Specific Encoding Algorithm (SeqPose) of Nucleotide Sequences and Its Application for Detecting Enhancers. *Int. J. Mol. Sci.* **2021**, *22*, 3079. [[CrossRef](#)]
59. Niu, K.; Luo, X.; Zhang, S.; Teng, Z.; Zhang, T.; Zhao, Y. iEnhancer-EBLSTM: Identifying Enhancers and Strengths by Ensembles of Bidirectional Long Short-Term Memory. *Front. Genet.* **2021**, *12*, 385. [[CrossRef](#)]
60. Yang, R.; Wu, F.; Zhang, C.; Zhang, L. iEnhancer-GAN: A Deep Learning Framework in Combination with Word Embedding and Sequence Generative Adversarial Net to Identify Enhancers and Their Strength. *Int. J. Mol. Sci.* **2021**, *22*, 3589. [[CrossRef](#)]
61. Khan, Z.U.; Pi, D.; Yao, S.; Nawaz, A.; Ali, F.; Ali, S. piEnPred: A bi-layered discriminative model for enhancers and their subtypes via novel cascade multi-level subset feature selection algorithm. *Front. Comput. Sci.* **2021**, *15*, 156904. [[CrossRef](#)]
62. Yang, H.; Wang, S.; Xia, X. iEnhancer-RD: Identification of enhancers and their strength using RKPK features and deep neural networks. *Anal. Biochem.* **2021**, *630*, 114318. [[CrossRef](#)] [[PubMed](#)]
63. Liang, Y.; Zhang, S.; Qiao, H.; Cheng, Y. iEnhancer-MFGBDT: Identifying enhancers and their strength by fusing multiple features and gradient boosting decision tree. *Math. Biosci. Eng.* **2021**, *18*, 8797–8814. [[CrossRef](#)] [[PubMed](#)]
64. Jumper, J.; Evans, R.; Pritzel, A.; Green, T.; Figurnov, M.; Ronneberger, O.; Tunyasuvunakool, K.; Bates, R.; Žídek, A.; Potapenko, A. Highly accurate protein structure prediction with AlphaFold. *Nature* **2021**, *596*, 583–589. [[CrossRef](#)] [[PubMed](#)]
65. Baek, M.; DiMaio, F.; Anishchenko, I.; Dauparas, J.; Ovchinnikov, S.; Lee, G.R.; Wang, J.; Cong, Q.; Kinch, L.N.; Schaeffer, R.D. Accurate prediction of protein structures and interactions using a 3-track network. *bioRxiv* **2021**. [[CrossRef](#)]
66. Yu, L.; Zhang, W.; Wang, J.; Yu, Y. Seqgan: Sequence generative adversarial nets with policy gradient. In Proceedings of the AAAI Conference on Artificial Intelligence, San Francisco, CL, USA, 4–9 February 2017.
67. LeCun, Y.; Boser, B.; Denker, J.; Henderson, D.; Howard, R.; Hubbard, W.; Jackel, L. Handwritten digit recognition with a back-propagation network. *Adv. Neural Inf. Process. Syst.* **1989**, *2*, 396–404.
68. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**. [[CrossRef](#)]
69. Ernst, J.; Kheradpour, P.; Mikkelsen, T.S.; Shores, N.; Ward, L.D.; Epstein, C.B.; Zhang, X.; Wang, L.; Issner, R.; Coyne, M. Mapping and analysis of chromatin state dynamics in nine human cell types. *Nature* **2011**, *473*, 43–49. [[CrossRef](#)]
70. Ernst, J.; Kellis, M. ChromHMM: Automating chromatin-state discovery and characterization. *Nat. Methods* **2012**, *9*, 215–216. [[CrossRef](#)]
71. Li, W.; Godzik, A. Cd-hit: A fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics* **2006**, *22*, 1658–1659. [[CrossRef](#)]

72. Fu, L.; Niu, B.; Zhu, Z.; Wu, S.; Li, W. CD-HIT: Accelerated for clustering the next-generation sequencing data. *Bioinformatics* **2012**, *28*, 3150–3152. [[CrossRef](#)]
73. Huang, Y.; Niu, B.; Gao, Y.; Fu, L.; Li, W. CD-HIT Suite: A web server for clustering and comparing biological sequences. *Bioinformatics* **2010**, *26*, 680–682. [[CrossRef](#)] [[PubMed](#)]
74. Huang, G.; Zeng, W. A discrete hidden Markov model for detecting histone crotonyllysine sites. *Match Commun. Math. Comput. Chem.* **2016**, *75*, 717–730.
75. Puton, T.; Kozłowski, L.; Tuszyńska, I.; Rother, K.; Bujnicki, J.M. Computational methods for prediction of protein–RNA interactions. *J. Struct. Biol.* **2012**, *179*, 261–268. [[CrossRef](#)] [[PubMed](#)]
76. Huang, G.; Chu, C.; Huang, T.; Kong, X.; Zhang, Y.; Zhang, N.; Cai, Y.-D. Exploring mouse protein function via multiple approaches. *PLoS ONE* **2016**, *11*, e0166580. [[CrossRef](#)]
77. Huang, G.; Zheng, Y.; Wu, Y.-Q.; Han, G.-S.; Yu, Z.-G. An information entropy-based approach for computationally identifying histone lysine butyrylation. *Front. Genet.* **2020**, *10*, 1325. [[CrossRef](#)]
78. Liu, K.; Cao, L.; Du, P.; Chen, W. im6A-TS-CNN: Identifying the N6-methyladenine site in multiple tissues by using the convolutional neural network. *Mol. Ther. Nucleic Acids* **2020**, *21*, 1044–1049. [[CrossRef](#)]
79. Fang, T.; Zhang, Z.; Sun, R.; Zhu, L.; He, J.; Huang, B.; Xiong, Y.; Zhu, X. RNAm5CPred: Prediction of RNA 5-methylcytosine sites based on three different kinds of nucleotide composition. *Mol. Ther. Nucleic Acids* **2019**, *18*, 739–747. [[CrossRef](#)]
80. Li, J.; Li, H.; Ye, X.; Zhang, L.; Xu, Q.; Ping, Y.; Jing, X.; Jiang, W.; Liao, Q.; Liu, B. IIMLP: Integrated information-entropy-based method for LncRNA prediction. *BMC Bioinform.* **2021**, *22*, 243. [[CrossRef](#)]
81. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
82. Targ, S.; Almeida, D.; Lyman, K. Resnet in resnet: Generalizing residual architectures. *arXiv* **2016**. [[CrossRef](#)]
83. Li, S.; Jiao, J.; Han, Y.; Weissman, T. Demystifying resnet. *arXiv* **2016**. [[CrossRef](#)]
84. Siami-Namini, S.; Tavakoli, N.; Namin, A.S. The performance of LSTM and BiLSTM in forecasting time series. In Proceedings of the 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 9–12 December 2019; pp. 3285–3292.
85. Kiperwasser, E.; Goldberg, Y. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Trans. Assoc. Comput. Linguist.* **2016**, *4*, 313–327. [[CrossRef](#)]
86. Liu, G.; Guo, J. Bidirectional LSTM with attention mechanism and convolutional layer for text classification. *Neurocomputing* **2019**, *337*, 325–338. [[CrossRef](#)]
87. Hong, Z.; Zeng, X.; Wei, L.; Liu, X. Identifying enhancer–promoter interactions with neural network based on pre-trained DNA vectors and attention mechanism. *Bioinformatics* **2020**, *36*, 1037–1043. [[CrossRef](#)] [[PubMed](#)]
88. Neishi, M.; Sakuma, J.; Tohda, S.; Ishiwatari, S.; Yoshinaga, N.; Toyoda, M. A bag of useful tricks for practical neural machine translation: Embedding layer initialization and large batch size. In Proceedings of the 4th Workshop on Asian Translation (WAT2017), Taipei, Taiwan, 27 November 2017; pp. 99–109.
89. Allen-Zhu, Z.; Li, Y.; Song, Z. A convergence theory for deep learning via over-parameterization. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 10–15 June 2019; pp. 242–252.
90. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)]
91. Pearlmutter, B.A. Learning state space trajectories in recurrent neural networks. *Neural Comput.* **1989**, *1*, 263–269. [[CrossRef](#)]
92. Giles, C.L.; Kuhn, G.M.; Williams, R.J. Dynamic recurrent neural networks: Theory and applications. *IEEE Trans. Neural Netw.* **1994**, *5*, 153–156. [[CrossRef](#)]
93. Sak, H.; Senior, A.; Rao, K.; Beaufays, F. Fast and accurate recurrent neural network acoustic models for speech recognition. *arXiv* **2015**. [[CrossRef](#)]
94. Saha, S.; Raghava, G.P.S. Prediction of continuous B-cell epitopes in an antigen using recurrent neural network. *Proteins: Struct. Funct. Bioinform.* **2006**, *65*, 40–48. [[CrossRef](#)]
95. Arras, L.; Montavon, G.; Müller, K.-R.; Samek, W. Explaining recurrent neural network predictions in sentiment analysis. *arXiv* **2017**. [[CrossRef](#)]
96. Du, Y.; Wang, W.; Wang, L. Hierarchical recurrent neural network for skeleton based action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1110–1118.
97. Medsker, L.R.; Jain, L. Recurrent neural networks. *Des. Appl.* **2001**, *5*, 64–67.
98. Schuster, M.; Paliwal, K.K. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.* **1997**, *45*, 2673–2681. [[CrossRef](#)]
99. Raffel, C.; Ellis, D.P. Feed-forward networks with attention can solve some long-term memory problems. *arXiv* **2015**. [[CrossRef](#)]
100. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
101. Hinton, G.E.; Srivastava, N.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R.R. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv* **2012**. [[CrossRef](#)]
102. Baldi, P.; Sadowski, P.J. Understanding dropout. *Adv. Neural Inf. Processing Syst.* **2013**, *26*, 2814–2822.