MATCH
Communications in Mathematical and in Computer Chemistry

MATCH Commun. Math. Comput. Chem. **90** (2023) 537–559
ISSN: 0340–6253
doi: 10.46793/match.90-3.537Z

Fer-COCL: A Novel Method Based on Multiple Deep Learning Algorithms for Identifying Fertility–Related Proteins

Shengli Zhang[a], Xinjie Li[a], Hongyan Shi[a], Yuanyuan Jing[a], Yunyun Liang[b], Yusen Zhang[c,*]

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

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

[c]*School of Mathematics and Statistics, Shandong University, Weihai 264209, P. R. China*

zhangys@sdu.edu.cn

(Received December 26, 2022)

Abstract

The survival of species depends on the fertility of organisms. It is also worthwhile to study the proteins that can regulate the reproductive activity of organisms. Since biological experiments are laborious to confirm proteins, it has become a priority that develop relevant computational models to predict the function of fertility-related proteins. With the development of machine learning, pertinent various algorithms can be the key to identifying fertility-related proteins. In this work, we develop a model Fer-COCL based on deep learning. The model consists of multiple features as well as multiple deep learning algorithms. First, we extract features using Amino acid composition (AAC), Dipeptide composition (DPC), CTD transition (CTDT) and deviation between the dipeptide and the expected mean (DDE). After that, the spliced features are fed

*Corresponding author.

into the classifier. The data processed jointly by convolutional neural network and long short-term memory is input to the fully connected layer for classification. After evaluating the model using 10-fold cross-validation, the accuracy of the two data sets reaches 97.1% and 98.3%, respectively. The results indicate that the model is efficient and accurate, facilitating biologists' research on biological fertility. In addition, a free online tool for predicting the function of fertility-related proteins is available at http://fercocl.zhanglab.site/.

1 Introduction

Fertility is a critical factor in the survival and continuation of organisms. There are many factors that affect fertility, but the most important is fertility-related protein. For example, reproductive development events, including spermatogenesis and oogenesis, as well as various other differentiation processes, such as embryogenesis and organogenesis, are regulated by many protein signal cascades that are essential for normal development. In the first stage of sexual reproduction and after the fertilized egg stage, there are many proteins involved. Because proteins are associated with fertility, their identification on a large scale will lead to a detailed understanding of the biological processes of oogenesis and spermatogenesis. Therefore, the study of fertility-related proteins has been a significant focus of biologists. During biological development, fertility-related proteins are also involved in regulating life activities [1]. The identifying of fertility-related proteins can help discover the underlying mechanisms of biological fertility processes and reveal their complex relationship with life activities [2]. Moreover, the study of fertility-related proteins has facilitated the development of drugs targeting infertility.

There have been various studies on related proteins. Huang [3] et al. found that Proteasome activators (PA28 and PA200) have an essential role in male fertility. By combining immunoblotting and two-dimensional gel electrophoresis, Schumacher [4] et al. discovered a variety of human sperm proteins, which contributed to the study of the relationship between protein evolution and phosphorylation status in mammalian sperm. Chen [5] et al. used the shotgun method to identify 246 new proteins in the reproductive organs of silkworm larvae, explaining the relationship between sexual dimorphism and proteins in germ cells. These show that the correct recognition of fertility-related proteins is inseparable from the research of their associated organismal activities.

Because it is time and effort-consuming to determine protein function through biological experiments, researchers have to develop novel models to predict proteins from the sequence of their amino acid composition and structure information. Bakhtiarizadeh [6] et al. trained a two-layer model PrESOgenesis based on support vector machine to recognize fertility-related proteins. The first layer determines if the protein is associated with fertility, and the second layer determines the protein's ability to assess fertility. Le [7] et al. used a deep learning algorithm to identify fertility-related proteins. They constructed the model Fertility-GRU using gated recurrent units. Wang [8] et al. proposed a model called Fertility-LightGBM, which used Lasso to filter the extracted features and then used LightGBM to classify the proteins. However, the above methods still need to improve the final prediction results due to the limitations of features and classifiers.

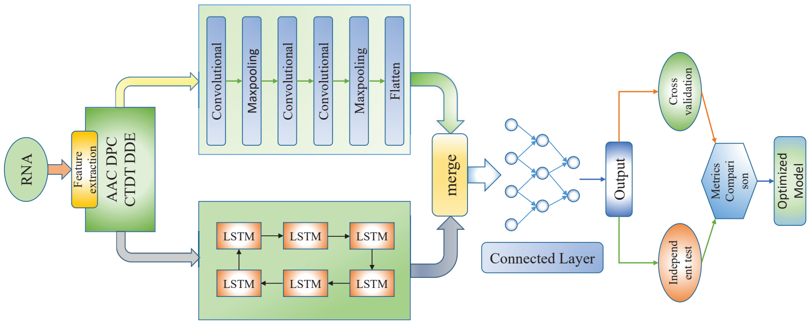


Figure 1. The operating flow of Fer-COCL.

To overcome the previous drawbacks, we construct a new model for predicting fertility-related proteins. First, sequence features are extracted using multiple feature representation methods, and the features contain information on the position of amino acids, amino acid composition and physicochemical properties. Next, the combined features are input to a classifier. The classifier contains two branches, the first branch consists of a convolutional neural network, and the second branch uses long short-term memory. Both branches process the data simultaneously, after which the data are fused and sent to the fully connected layer for classification. Finally, this model is evaluated, and the evaluation results are compared with previous methods. Due to the comprehensiveness of the protein features and the integration of deep learning algorithms, better results than the previous ones are finally achieved. Figure 1 shows the flow chart of the model.

2 Materials and methods

2.1 Dataset

To develop accurate predictive models, a usable and objective benchmark dataset is first required. The training data needs to be as complete and accurate as possible to optimize the prediction performance of the model. Bakhtiarizadeh [6] et al. created a raw positive dataset of proteins by searching the UniProt Knowledge Base (UniProtKB) [9] and excluded proteins with sequences greater than 6000 or less than 60. Then, they removed pairs of sequences with more than 50% similarity by CD-HIT [10] and deleted sequences containing non-canonical residues ("B", "X" or "Z"). Sequence similarity was obtained by calculating the number of similar amino acid residues at the corresponding positions of two equal-length sequences with the same residues and the percentage of the total length. After a series of manipulations, 1704 proteins were finally obtained and used as positive samples. 1815 non-fertility proteins were also obtained by a similar method by Le [7] et al. In this study, we use the same data as Bakhtiarizadeh [6] and Le [7], and split the initial dataset according to 8:2, with eight as the training set, and two as the independent test set. The datasets are shown in Table 1.

Table 1. The number of training sets and independent test sets.

	ALL	Training sets	Independent test sets
Fertility	1704	1420	284
Non-fertility	1815	1512	303

2.2 Feature extraction

The feature representation of biological sequence is to represent sequence information with digital information, which is an essential step for model building [11]. Our model uses the following four feature representation methods to fully extract protein sequence information in terms of protein location, amino acid composition and physicochemical properties.

2.2.1 Amino acid composition

AAC [12] is a feature representation based on the amino acid position as well as the composition of a protein that depicts the local information of the sequence. The AAC of each sequence can be calculated according to the composition of 20 amino acids in the sequence. The following is the specific calculation:

Given a protein sequence P of length L:

$$P = [\alpha_1, \alpha_2, \alpha_3, ..., \alpha_{20}] \tag{1}$$

where α_i represents the amount of occurrence of the i-th amino acid composition, and the equation is as follows:

$$\alpha_i = \frac{\varphi_i}{L} \tag{2}$$

where φ_i denotes the number of occurrences of the i-th amino acid composition, and L denotes the protein sequence length.

In this work, a protein sequence P is denoted as a 20-dimensional vector.

2.2.2 Dipeptide composition

Dipeptide composition (DPC) [13–15] is a feature representation method to extract the position information of dipeptides, and usually calculates the frequency of amino acid pairs. DPC fully considers the coupling of adjacent amino acids and the position information of amino acid residues. There are 400 types of dipeptides as $\{AA, AC, AD, \cdots, YY\}$. Therefore, for the protein sequence P, DPC is encoded as follows:

$$\phi = (\varepsilon_1, \varepsilon_2, \varepsilon_3, ..., \varepsilon_{400}) \tag{3}$$

where ε_i is the amount of appearances of the i-th dipeptide composition.

2.2.3 CTD transition

CTD transition (CTDT) [16] is a feature representation method that combines the amino acid distribution with the physicochemical properties of amino acid residues. The method divides amino acid residues into three clusters according to their physicochemical properties. CTD [17] describes the information on the location of amino acids and their global and local physicochemical properties. The thirteen physicochemical properties are Hydrophobicity_ PRAM900101, Hydrophobicity_ ARGP820101, Hydrophobicity_ ZIMJ680101, Hydrophobicity_ PONP930101, Hydrophobicity_ CASG920101, Hydrophobicity_ ENGD860101, Hydrophobicity_ FASG890101, Normalized van der Waals volume, Polarity, Polarizability, Charge, Secondary structure, Solvent accessibility. C represents the information on amino acid grouping; T is the information on the frequency of dipeptides; and D is the information on the frequency and position of the first, 25%, 50%, 75% and last occurrence of histone sequences. The three types of information are calculated as follows:

$$C(r) = \frac{N(r)}{N} \tag{4}$$

$$T(r,s) = \frac{N(r,s) + N(s,r)}{N-1} \tag{5}$$

$$D(r) = \left(\frac{M(r,1)}{N}, \frac{M(r,2)}{N}, \frac{M(r,3)}{N}, \frac{M(r,4)}{N}, \frac{M(r,5)}{N}\right) \tag{6}$$

where $r \in$ {positive, neutral, negative}, $(r,s) \in$ {(positive, neutral), (neutral, negative), (negative, positive)}, $N(r)$ is the magnitude of the r-th group in the amino acid, and $N(r,s)$ denotes the frequency of occurrence of the dipeptide component from the r-th group to the s-th group. $M(r,i)$ denotes the position information of the r-th moiety at the i-th position.

Based on the information obtained from the CTD, the final CTDT formula is expressed as:

$$CTDT(r,s) = \frac{N(r,s)+N(s,r)}{N} \tag{7}$$

Based on the 13 physicochemical properties of amino acid residues, a sequence is finally represented as a 39-dimensional vector.

2.2.4 Deviation between the dipeptide and the expected mean

Deviation between the dipeptide and the expected mean (DDE) [18] is a method for extracting features from a protein sequence based on the dipeptide composition and the theoretical mean and theoretical variance of the encoders. First, three parameters are dipeptide composition, mean and variance of theoretical, respectively. In the i-th dipeptide of the protein sequence, the three parameters are calculated as follows:

$$\begin{cases} DC(i) = \frac{a_i}{P-1} \\ TM(i) = \frac{C_{i1}}{C_P} * \frac{C_{i2}}{C_P} \\ TV(i) = \frac{1-TM(i)}{P-1} \end{cases} \tag{8}$$

where a_i represents the frequency of the i-th dipeptide, P denotes the sequence length, C_{i1} is the number of codes encoding the first amino acid, C_{i2} is the number of codes encoding the second amino acid, and C_P is the number of all possible codons after removing the three termination encoders. Based on the three parameters obtained from the calculation, the DDE is defined as:

$$DDE(i) = \frac{DC(i)-TM(i)}{\sqrt{TV(i)}} \tag{9}$$

For a protein sequence P, the DDE approach to encoding yields the following equation:

$$DDE = \{DDE_1, DDE_2, DDE_3, ..., DDE_{400}\} \tag{10}$$

2.3 Multiple deep learning algorithms

Deep learning, a prevalent method nowadays, has a wide range of applications in many fields, such as speech recognition [19], automatic machine translation [20], autonomous driving [21], instant visual translation [22], and so on. To raise the accurate of model prediction, we present a deep learning-based fusion classifier that combines the advantages of convolutional neural networks (CNN) [23] and long short-term memory (LSTM) [24] with efficient and accurate properties. It contains two branches; the first branch contains three convolutional layers, two pooling layers, a dropout layer, a flat layer and a fully connected layer; the second branch contains an LSTM with 16 neurons. After the two branches are computed separately, the results are fused and fed into four fully connected layers to obtain the output finally. The digital vectors encoded by AAC, DC, CTDT and DDE are combined as input features.

In the convolution layer [25], In the convolution layer, the various parameters of the convolution can have a great impact on the experiment. In order to obtain the optimal results, the number of convolutional kernels per layer is set to 128, 64, 32, 16, 8, and the size of convolutional kernels are set to 1*4, 1*3, 1*2. Different combinations of convolutional layers are tested, and finally the convolutional layers with the number of convolutional kernels 64, 32, 16, and the size of 1*3, 1*2 are selected. Meanwhile, to facilitate the extraction of more features, the convolution kernel step size is set to 1. After the convolution operation, Relu and LeakyRelu are applied as activation functions, respectively, which are the most commonly used activation functions in deep learning. The computational equations in the convolution layer are as follows:

$$Conv(x) = \sum_{p=1}^{P} \sum_{q=1}^{Q} w_{pq}^{k} x_{i+p,q} \tag{11}$$

$$\mathrm{Relu} = \max(x, 0) \tag{12}$$

$$Leaky\mathrm{Rel}u(x) = \begin{cases} x & if \quad x > 0 \\ \alpha x & if \quad x < 0 \end{cases} \tag{13}$$

where w represents the size of the convolution kernel, $W^k = (w^k_{pq})_{p*q}$ represents the weight matrix of the i-th convolution kernel, and the scale of the matrix is P*Q.

In order to achieve high accuracy and calculation speed, a pooling layer is added after the convolutional layer. Pooling operations are divided into maximum pooling, average pooling and global pooling. After testing, we can get the best result by choosing the maximum pooling. The maximum pooling operation selects the largest number among the data selected in the sliding window as the pooling output. After debugging, the pooling range is set to 1*2 and the step size is 1 to get the best results. The formula for the maximum pooling operation is as follows:

$$pool(x)_i = \max(x_1, x_2, x_3, ..., x_n)_i \tag{14}$$

where i is the i-th pooling operation, and x_i is the number in the sliding window.

The dropout layer [26] is essential to prevent overfitting of the model while enhancing generalization. It can randomly select some neurons to stop working probabilistically. We increase the dropout rate from 0.1 to 0.9 for experiments respectively. When the dropout rate is set to 0.5, it can efficiently protect against overfitting and improve the accuracy of prediction at the same time. Then the flat layer is used to perform a smoothing operation to get the output result of the first branch.

In the second branch, we use long short-term memory [27] to process data. As an improvement of the recurrent neural network, long short-term memory has the characteristics of remembering important information. It contains four critical units: input gate, cell state, forget gate and output gate. First, the forget gate [28] judges the input features and chooses which information to keep or forget. The equation for the forget gate is as follows:

$$f_t = \sigma(W_f \; * \; [h_{t-1}, x_t] + b_f) \tag{15}$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{16}$$

where x_t represents the input information at time t, W_f is the weight matrix of the forget gate, f_{t-1} is the output at the previous time, and b is the offset value of each unit.

The information selected through the forget gate is input into the input gate and the cell state. The sigmoid layer of the input gate [29] determines the updated value. Then, candidate values are generated through the tanh layer and appended to the cell state. The specific equation is as follows:

$$i_t = \sigma(W_i \ * \ [h_{t-1}, x_t] + b_i) \tag{17}$$

$$\widetilde{C}_t = \tanh(W_C \ * \ [h_{t-1}, x] + b_C) \tag{18}$$

where i_t represents the output information of the input gate at time t, $\widetilde{C}_t$ is the temporary state.

Immediately following updating the cell state [30], after the update is completed, the cell state is multiplied by the output information of the sigmoid layer through the tanh layer, and finally the required output is obtained. The calculation of the output gate is as follows:

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t \tag{19}$$

$$o_t = \sigma(W_o \ * \ [h_{t-1}, x_t] + b_o) \tag{20}$$

$$h_t = o_t * \tanh(C_t) \tag{21}$$

where h_t is the final output at time t.

In order to fuse with the output of the convolution, we set the number of cells of the LSTM to 16. The activation functions are kept at their default settings, i.e. the most commonly used tanh and sigmoid functions. Since the number of LSTM iterations is small, the dropout rate is set to 0.

After the two branches are calculated separately, the output data is merged. The fusion data is input into four fully connected layers [31], and the numbers of their neurons are 128, 64, 32, and 2, respectively. Finally, the sigmoid function is used for processing, and the classification result is obtained.

2.4 Model evaluation

To observe the accuracy and robustness of the model, cross-validation is considered the most efficient method, which consists of jackknife test, independent dataset test and k-fold cross-validation [32–40]. In this work, 10-fold cross-validation is used to validate the baseline dataset, and the independent dataset is used to validate the general applicability of the model. At the same time, four valid assessment indexes are used to indicate the performance of the evaluated models, namely Accuracy (ACC), Sensitivity (SN), Specificity (SP), and Mathews Correlation Coefficient (MCC) [41–44]. Accuracy should be the most basic evaluation index, which describes whether the prediction of the overall result is correct or not. Sensitivity describes the percentage of all identified positive samples among all positive samples. Specificity describes the percentage of all identified negative samples among all negative samples. They are defined as follows:

$$\begin{aligned} &Sn = \tfrac{TP}{TP+FN} \\ &SP = \tfrac{TN}{TN+FP} \\ &ACC = \tfrac{TP+TN}{TP+TN+FP+FN} \\ &MCC = \tfrac{TP\times TN-FP\times FN}{\sqrt{(TP+FN)(TP+FP)(TN+FP)(TN+FN)}} \end{aligned} \tag{22}$$

where TP, TN, FP and FN represent the number of true positive, true negative, false positive, and false negative, respectively. ROC curves [45] can examine a classifier's ability to recognize samples at a certain threshold. The horizontal coordinate of the curve is the sensitivity and the vertical coordinate represents the 1-specificity. Generally speaking, if the ROC is smooth, it can basically be judged that there is not too much over-fitting. At this time, the model can only look at the area enclosed by the curve and the coordinate axis. The larger the area of the curve, the better the prediction of the model.

3 Result and discussion

3.1 Comparison of single feature extraction methods

Feature extraction methods play a crucial role in prediction models by converting text sequences into digital sequences, which provide input data for the subsequent classifier. Various feature representation methods are able to gain different information about biological sequences from different perspectives. However, the information that can be obtained from a single feature approach is not comprehensive enough, so we use a multi-feature extraction method to represent protein sequences.

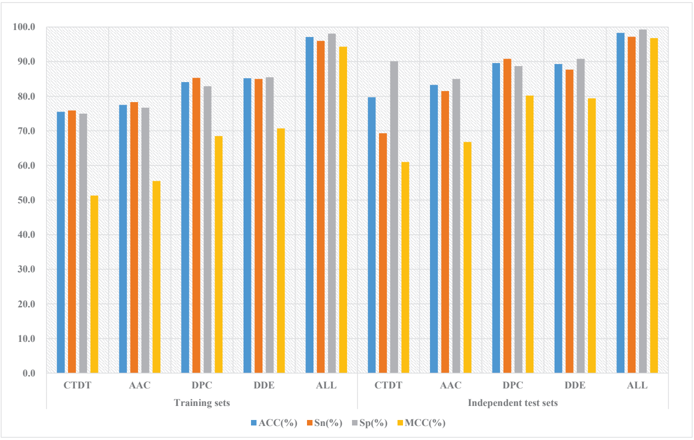


Figure 2. Comparison of performance between single feature and full feature fusion for different datasets.

In this work, we used AAC, DPC, CTDT and DDE to extract sequence information in terms of protein sequence composition, amino acid positions and physicochemical properties. To illustrate the need for a multi-feature extraction approach, we combined the four features to obtain new features, which were fed into our classifier CO-CLS for classification prediction. At the same time, we separately input single features to the same classifier for prediction. Afterwards, the results of the combined features are compared with the results of the single features. The comparison results are represented in Figure 2, where ALL indicates the new features after combination. It can be clearly seen that all evaluation metrics of the new features exceed those of the single features. The accuracy rates of 97.1% and 98.3% for the two datasets were achieved, which is a great improvement.

The experimental results of single features are inferior to those of combined features because the feature information obtained from a single feature extraction method is less, resulting in too low data dimensionality for the input classifier, which is not conducive to learning by deep learning classifiers. Meanwhile, the single aspect of the biometric feature information cannot represent the biological sequence comprehensively, and there is some information missing, which affects the final experimental results. This suggests that the combination of multi-feature methods can be of great help to the experiments.

3.2 Comparison of multiple feature extraction methods

In order to select the optimal way to combine the features, we combine three features and end up with four combined features. Both these four copies of features and the new features obtained by combining all the features are fed into the same classifier CO-CLS for experiments. The obtained results are plotted in a table, and Table 2 shows the comparison results. The table shows that in the experiments where the three features are combined, the ACC and MCC of the model are greatly reduced. This indicates that the accuracy and reliability of the model prediction results are reduced. In contrast, the best experimental results were obtained by combining all the features, and all the indicators exceeded the results of any one feature combination.

Table 2. Performance comparison of multiple feature extraction methods on training sets and independent test sets.

Data	Feature	ACC(%)	Sn(%)	Sp(%)	MCC(%)	AUC
Training sets	CTDT+DPC+AAC	89.3	89.7	88.9	78.6	0.947
	CTDT+DDE+AAC	89.7	88.2	91.2	79.6	0.954
	DPC+DDE+AAC	90.9	92.3	89.7	82.2	0.958
	CTDT+DPC+DDE	92.9	94.4	91.5	85.9	0.944
	ALL	97.1	96.0	98.1	94.3	0.989
Independent test sets	CTDT+DPC+AAC	92.7	90.8	94.5	85.5	0.970
	CTDT+DDE+AAC	92.8	92.5	93.2	85.8	0.978
	DPC+DDE+AAC	92.2	91.5	92.8	84.4	0.976
	CTDT+DPC+DDE	94.7	93.2	96.2	89.8	0.977
	ALL	98.3	97.2	99.3	96.8	0.994

This indicates that any one feature is essential in this work. Four feature extraction methods extracted four aspects of biological features. These biological features represent the biological information about the position, composition, physicochemical properties of amino acids, and statistical orientation of the sequences. The four features are combined to better represent the whole biological sequence. In addition, the combination of all features increases the amount of input data, which is more conducive for the model to perform learning and improve the results of the experiment. Therefore, the combination of all features is the optimal choice for the model.

3.3 Comparison with different classifiers

A classifier is an important tool for learning the input data and is an integral and important part of the model. With the continuous development and updating of algorithms, many practical classifiers have emerged, including traditional classifiers such as support vector machines (SVM) [46], random forests (RF) [47], and logistic regression (LR) [48]; and increasingly popular deep learning classifiers such as deep neural networks (DNN) [49], convolutional neural networks (CNN) [50], recurrent neural networks (RNN) [51], and so on. To choose one of the best classifiers, we experiment with the classifiers listed above, including tree models, non-tree models, and deep learning. These representative algorithms can help us quickly find algorithms that are more conducive to identifying fertility-related proteins. The parameters used by the classifiers are shown in Table 3. In our experiments, we fuse two types of deep learning to form a new model, which is a new classifier that combines convolutional neural networks with long short-term memory.

Table 3. The parameters of different classifiers.

Classifier	Parameter	
SVM	probability=True, kernel="poly"	
RF	n_estimators=500, criterion="gini", max_depth=10	
DNN	Four Dense layers with 16, 8 and 4 neurons respectively	activation = "sigmoid", loss = categorical_crossentropy, optimizer = Adam, metrics = ["accuracy"]
CNN	Two Conv1D layer and a MaxPooling1D layer	
LSTM	Four layers of LSTM with 64, 32, 16 and 8 neurons respectively	
CO-CLS	Parameter combination of CNN and LSTM	

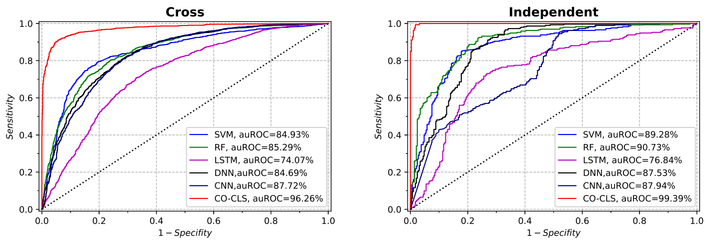
Table 4. Comparative performance with different classifiers.

Data	Classifier	ACC(%)	Sn(%)	Sp(%)	MCC(%)
Training sets	SVM	82.3	78.5	86.0	65.0
	RF	77.6	74.1	81.2	55.4
	LSTM	64.1	53.3	74.9	41.0
	DNN	89.5	93.7	85.6	79.5
	CNN	82.2	84.0	80.4	65.3
	CO-CLS	97.1	96.0	98.1	94.3
Independent test sets	SVM	79.3	77.2	81.5	58.9
	RF	82.4	79.5	85.3	65.3
	LSTM	70.3	67.2	73.4	43.5
	DNN	84.4	82.8	86.0	69.0
	CNN	76.6	75.7	77.4	58.3
	CO-CLS	98.3	97.2	99.3	96.8

We use the method of controlling variables to ensure that all classifiers have the same input features. This can minimize the interference of external factors. Table 4 and Figure 3 show the comparison results of different classifiers. Obviously, the best results are obtained by our classifier, with AUC values of 0.989, 0.994 for the two data sets, respectively. Compared to traditional classifiers, our model uses deep learning classifiers, which have the advantages of fast computation and good robustness. Compared to a single deep learning model, we use a new model that incorporates two single classifiers. CNN can quickly learn more useful features and prevent overfitting; LSTM can memorize important information and ignore useless information, and can also prevent the risk of gradient disappearance or gradient explosion. The combination of the two algorithms ensures the efficiency and accuracy of the model.

3.4 Comparison with existing methods

To estimate the properties of our model, we compared it with several previously published articles. The Figure 4 shows the results of its comparison with other models on two datasets. On both datasets, the results achieved by our model largely outperform the results of other models, with ACC values 8.6% and 6.8% higher than the previous optimal ones, respectively. Other metrics also improved significantly compared to previous articles.



(a) Training set (b) Independent set

Figure 3. Comparison of ROC curve for different classifiers.

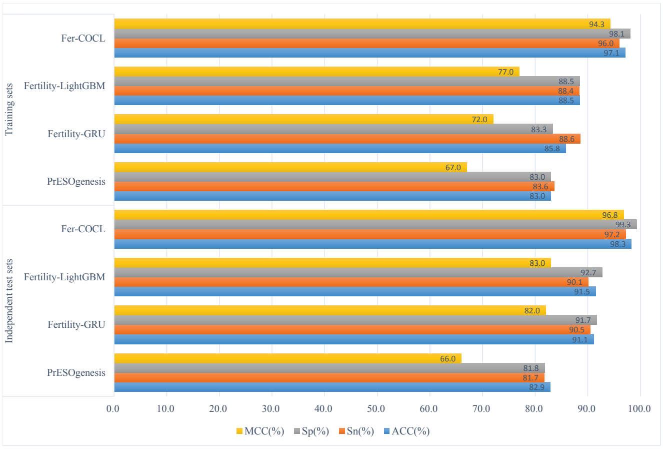


Figure 4. Comparison of performance for existing methods.

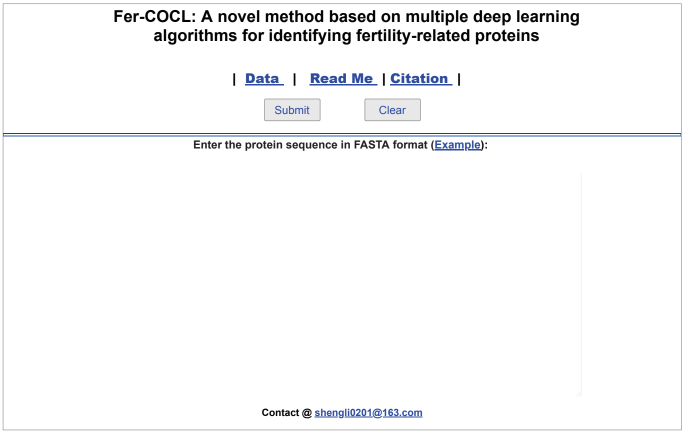
Compared with other models, our model has two advantages. First, we consider extracting features from different perspectives, including the positional information of sequences, component information, physicochemical properties, and statistical information, and obtain more comprehensive features than before. Secondly, we use deep learning algorithms as classifiers. By comparing with SVM, RF, XGBoost, it is able to perform faster learning while improving the accuracy of the model prediction.

Compared with other deep algorithms, our model also has corresponding advantages. First, we use a convolutional neural network for representation learning, extracting features by convolution quadratic and nonlinear processing by two different activation functions, Relu and LeakyRelu, to improve the accuracy of the model. Then, we process the data using long and short-term memory, which is capable of processing time series, filtering out useless information and remembering the important ones. The combination of two deep learning algorithms is more advantageous than a single algorithm. Therefore, it can be widely used to predict fertility-related proteins.

4 Webserver

To make it convenient to use our model, we have developed a free to use webserver to improve usability [52]. Researchers can use the webserver to obtain prediction results quickly without complicated operations. Here, we give a guide to use the webserver.

The webserver is accessed from http://fercocl.zhanglab.site/, and the entry screen is shown in the Figure 5. The datasets of fertility-related proteins are available in Data. The detailed description is in Read Me module. The users simply enter the protein sequence to be predicted in the block and then click Submit to obtain the forecast results. It is important to note that the sequence entered must be in FASTA format.


Fer-COCL: A novel method based on multiple deep learning algorithms for identifying fertility-related proteins

| Data | Read Me | Citation |

Enter the protein sequence in FASTA format (Example):

Contact @ shengli0201@163.com

Figure 5. The webserver prediction interface for Fer-COCL.

5 Conclusion

Fertility proteins have an essential role in the evolution and reproduction of organisms. Therefore, accurate predictions of fertility proteins can largely aid biologists in their research. Currently, many models have been proposed, but the prediction of fertility proteins is not accurate enough. Therefore, we present a model named Fer-COCL based on a deep fusion algorithm. We utilize multiple methods to present protein features from various perspectives, and use convolutional neural networks and long short-term memory fusion as classifiers to achieve better results. The accuracy on the two datasets reached 97.1% and 98.3%, respectively, which is a huge improvement. The model can be built to better facilitate biological research.

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

References

[1] X. Y. Ng, B. A. Rosdi, S. Shahrudin, Prediction of antimicrobial peptides based on sequence alignment and support vector machine-pairwise algorithm utilizing LZ-complexity, *Biomed. Res. Int.* (2015) #212715.

[2] A. Rahman, R. Abdullah, W. Wan-Khadijah, Gametogenesis, fertilization and early embryogenesis in mammals with special reference to goat: A review, *J. Biol. Sci.* **8** (2008) 1115–1128.

[3] L. Huang, K. Haratake, H. Miyahara, T. Chiba, Proteasome activators, PA28 and PA200, play indispensable roles in male fertility, *Sci. Rep.* **6** (2016) #23171.

[4] J. Schumacher, S. Ramljak, A. R. Asif, M. Schaffrath, H. Zischler, H. Herlyn, Evolutionary conservation of mammalian sperm proteins associates with overall, not tyrosine, phosphorylation in human spermatozoa, *J. Proteome Res.* **12** (2013) 5370–5382.

[5] J. Chen, J. Li, Z. You, L. Liu, J. Liang, Y. Ma, M. Chen, H. Zhang, Z. Jiang, B. Zhong, Proteome analysis of silkworm, bombyx mori, larval

gonads: Characterization of proteins involved in sexual dimorphism and gametogenesis, *J. Proteome Res.* **12** (2013) 2422–2438.

[6] M. R. Bakhtiarizadeh, M. Rahimi, A. Mohammadi-Sangcheshmeh, V. Shariati J, S. A. Salami, PrESOgenesis: A two-layer multi-label predictor for identifying fertility-related proteins using support vector machine and pseudo amino acid composition approach, *Sci. Rep.* **8** (2018) #9025.

[7] N. Q. K. Le, Fertility-GRU: identifying fertility-related proteins by incorporating deep gated recurrent units and original position-specific scoring matrix profiles, *J. Proteome Res.* **18** (2019) 3503–3511.

[8] M. H. Wang, L. L. Yue, X. H. Yang, X. L. Wang, Y. Han, B. Yu, Fertility-LightGBM: A fertility-related protein prediction model by multi-information fusion and light gradient boosting machine, *Biomed. Signal Process. Control* **68** (2021) #102630.

[9] U. Consortium, UniProt: a hub for protein information, *Nucleic Acids Res.* **43** (2014) D204–D212.

[10] S. L. Zhang, J. Y. Wang, X. J. Li, Y. Y. Liang, M6A-GSMS: Computational identification of N6-methyladenosine sites with GBDT and stacking learning in multiple species, *J. Biomol. Struct. Dyn.* **40** (2022) 12380–12391.

[11] M. H. Wang, X. W. Cui, S. Li, X. H. Yang, A. J. Ma, Y. S. Zhang, B. Yu, DeepMal: Accurate prediction of protein malonylation sites by deep neural networks, *Chemometr. Intell. Lab. Sys.* **207** (2020) #104175.

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

[13] P. M. Feng, H. Lin, W. Chen, Identification of antioxidants from sequence information using Naïve Bayes, *Comput. Math. Methods Med.* **2** (2013) #567529.

[14] K. Ahmad, M. Waris, M. Hayat, Prediction of protein submitochondrial locations by incorporating dipeptide composition into Chou's general pseudo amino acid composition, *J. Membr. Biol.* **249** (2016) 293–304.

[15] H. Zhou, C. Chen, M. Wang, Q. Ma, B. Yu, Predicting Golgi–Resident protein types using conditional covariance minimization with XGBoost based on multiple features fusion, *IEEE Access* **7** (2019) 144154–144164.

[16] C. Wang, Q. Zou, A machine learning method for differentiating and predicting human-infective coronavirus based on physicochemical features and composition of the spike protein, *Chinese J. Electron.* **30** (2021) 815–823.

[17] Z. You, L. Zhu, C. Zheng, H. Yu, S. Deng, Z. Ji, Prediction of protein–protein interactions from amino acid sequences using a novel multi–scale continuous and discontinuous feature set, *Bioinformatics* **15** (2019) #S9.

[18] V. Saravanan, N. Gautham, Harnessing computational biology for exact linear B-cell epitope prediction: A novel amino acid composition-based feature descriptor, *Omics* **19** (2015) 648–658.

[19] J. Guglani, A. N. Mishra, DNN based continuous speech recognition system of Punjabi language on Kaldi toolkit, *Int. J. Speech Tech.* **24** (2021) 1–5.

[20] E. Comelles, J. Atserias, VERTa: a linguistic approach to automatic machine translation evaluation, *Lang. Res. Eval.* **53** (2019) 57–86.

[21] H. F. Liu, X. F. Han, X. R. Li, Y. Z. Yao, P. Huang, Z. M. Tang, Deep representation learning for road detection using Siamese network, *Multimedia Tools Appl.* **78** (2019) 24269–24283.

[22] M. Oh, L. Q. Zhang, DeepMicro: deep representation learning for disease prediction based on microbiome data, *Sci. Rep.* **10** (2020) #6026.

[23] M. Tahir, H. Tayara, K. T. Chong, iPseU-CNN: Identifying RNA pseudouridine sites using convolutional neural networks, *Mol. Therapy Nucleic Acids* **16** (2019) 463–470.

[24] J. W. Li, Y. Q. Pu, J. J. Tang, Q. Zou, F. Guo, DeepATT: a hybrid category attention neural network for identifying functional effects of DNA sequences, *Brief. Bioinf.* **22** (2020) #bbaa159.

[25] G. Aoki, Y. Sakakibara, Convolutional neural networks for classification of alignments of non-coding RNA sequences, *Bioinf.* **34** (2018) i237–i244.

[26] B. T. Yang, F. Liu, C. Ren, Z. Y. Ouyang, Z. W. Xie, X. C. Bo, W.J. Shu, BiRen: predicting enhancers with a deep-learning-based model using the DNA sequence alone, *Bioinf.* **33** (2017) 1930–1936.

[27] X. Y. Pan, P. Rjinbeek, J. C. Yan, H. B. Shen, Prediction of RNA-protein sequence and structure binding preferences using deep convolutional and recurrent neural networks, *BMC Bioinf.* **19** (2018) #511.

[28] W. Z. Lu, Y. Tang, H. J. Wu, H. M. Huang, Q. M. Fu, J. Qiu, H. O. Li, Predicting RNA secondary structure via adaptive deep recurrent neural networks with energy-based filter, *BMC Bioinf.* **20** (2019) #684.

[29] Q. X. Xiao, W. L. Li, Y. Z. Kai, P. Chen, J. Zhang, B. Wang, Occurrence prediction of pests and diseases in cotton on the basis of weather factors by long short term memory network, *BMC Bioinf.* **20** (2019) #688.

[30] C. H. Kang, U. Erdenebayar, J. Park, K. J. Lee, Multi-class classification of sleep apnea/hypopnea events based on long short-term memory using a photoplethysmography signal, *J. Med. Sys.* **44** (2019) 1654–1662.

[31] B. Yu, Z. M. Yu, C. Chen, A. J. Ma, B. Q. Liu, B. G. Tian, Q. Ma, DNNAce: Prediction of prokaryote lysine acetylation sites through deep neural networks with multi-information fusion, *Chemometr. Intell. Lab. Sys.* **200** (2020) #103999.

[32] Y. H. Huo, L. H. Xin, C. Z. Kang, M. H. Wang, Q. Ma, B. Yu, SGL-SVM: A novel method for tumor classification via support vector machine with sparse group Lasso, *J. Theor. Biol.* **48** (2020) #110098.

[33] X. M. Sun, T. Y. Jin, C. Chen, X. W. Cui, Q. Ma, B. Yu, RBPro-RF: Use Chou's 5-steps rule to predict RNA-binding proteins via random forest with elastic net, *Chemometr. Intell. Lab. Sys.* **197** (2020) #103919.

[34] S. L. Zhang, H. J. Qiao, KD-KLNMF: Identification of lncRNAs subcellular localization with multiple features and nonnegative matrix factorization, *Anal. Biochem.* **610** (2020) #113995.

[35] C. Chen, Q. M. Zhang, B. Yu, Z. M. Yu, P.J . Lawrence, Q. Ma, Y. Zhang, Improving protein-protein interactions prediction accuracy using XGBoost feature selection and stacked ensemble classifier, *Comput. Biol. Med.* **123** (2020) #103899.

[36] S. L. Zhang, F. Zhu, Q. H. Yu, X. Y. Zhu, Identifying DNA-binding proteins based on multi-features and LASSO feature selection, *Biopolymers* **112** (2021) #e23419.

[37] C. Chen, Q. M. Zhang, Q. Ma, B. Yu, LightGBM-PPI: predicting protein-protein interactions through LightGBM with multi-information fusion, *Chemometr. Intell. Lab. Sys.* **191** (2019) 54–64.

[38] J. Jia, Z. Liu, X. Xiao, B. Liu, K. C. Chou, iPPI-Esml: An ensemble classifier for identifying the interactions of proteins by incorporating their physicochemical properties and wavelet transforms into PseAAC, *J. Theor. Biol.* **377** (2015) 47–56.

[39] L. Y. Wei, S. S. Luan, L. A. E. Nagai, R. Su, Q. Zou, Exploring sequence based features for the improved prediction of DNA N4-methylcytosine sites in multiple species, *Bioinf.* **35** (2019) 1326–1333.

[40] J. Y. Wang, S. L. Zhang, H. J. Qiao, J. S. Wang, UMAP-DBP: An improved DNA-binding proteins prediction method based on uniform manifold approximation and projection, *Protein J.* **40** (2021) 562–575.

[41] R. Su, J. Hu, Q. Zou, B. Manavalan, L. Y. Wei, Empirical comparison and analysis of web-based cell-penetrating peptide prediction tools, *Brief. Bioinf.* **21** (2020) 408–420.

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

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

[44] B. Manavalan, R. G. Govindaraj, T. H. Shin, M. O. Kim, G. Lee, iBCE-EL: a new ensemble learning framework for improved linear B-cell epitope prediction, *Front. Immunol.* **9** (2018) #1695.

[45] Z. X. Zhao, X. C. Zhang, F. Chen, L. Fang, J. Y. Li, Accurate prediction of DNA N4-methylcytosine sites via boost-learning various types of sequence features, *BMC Genomics* **21** (2020) #627.

[46] C. Chen, Q. M. Zhang, B. Yu, Z. M. Yu, P. J. Lawrence, Q. Ma, Y. Zhang, Improving protein-protein interactions prediction accuracy using XGBoost feature selection and stacked ensemble classifier, *Comput. Biol. Med.* **123** (2020) #103899.

[47] J. S. Wang, S. L. Zhang, PA-PseU: An incremental passive-aggressive based method for identifying RNA pseudouridine sites via Chou's 5-steps rule, *Chemometr. Intell. Lab. Sys.* **210** (2021) #104250.

[48] M. M. Zhu, G. Michael, MiPepid: MicroPeptide identification tool using machine learning, *BMC Bioinf.* **20** (2019) 685–696.

[49] Y. B. Xie, X. T. Luo, Y. P. Li, L. Chen, W. B. Ma, J. J. Huang, J. Cui, Y. Zhao, Y. Xue, Z. X. Zuo, J. Ren, DeepNitro: prediction of protein nitration and nitrosylation sites by deep learning, *Genom. Proteom. Bioinf.* **16** (2018) 294–306.

[50] Q. Liu, F. Xia, Q. J. Yin, R. Jiang, Chromatin accessibility prediction via a hybrid deep convolutional neural network, *Bioinf.* **34** (2018) 732–738.

[51] M. N. Hamid, I. Friedberg, Identifying antimicrobial peptides using word embedding with deep recurrent neural networks, *Bioinf.* **35** (2019) 2009–2016.

[52] T. Xue, S. L. Zhang, H. J. Qiao, i6mA-VC: A multi-classifier voting method for the computational identification of DNA N6-methyladenine sites, *Interdis. Sci.* **13** (2021) 413–425.