A Protein-Protein Interaction Extraction Approach Based on Large Pre-trained Language Model and Adversarial Training

Zhan Tang[1], Xuchao Guo[1], Zhao Bai[1], Lei Diao[1], Shuhan Lu[2] and Lin Li[1*]
[1] College of Information and Electrical Engineering, China Agricultural University
Beijing, 100083 China
[e-mail: tz_blues@163.com]
[2] School of Information, University of Michigan
Ann Arbor, MI 48109 USA
[e-mail: lilincau@126.com]
*Corresponding author: Lin Li

Received July 13, 2021; revised October 7, 2021; revised October 28, 2021; revised December 20, 2021; accepted February 16, 2022; published March 31, 2022

Abstract

Protein-protein interaction (PPI) extraction from original text is important for revealing the molecular mechanism of biological processes. With the rapid growth of biomedical literature, manually extracting PPI has become more time-consuming and laborious. Therefore, the automatic PPI extraction from the raw literature through natural language processing technology has attracted the attention of the majority of researchers. We propose a PPI extraction model based on the large pre-trained language model and adversarial training. It enhances the learning of semantic and syntactic features using BioBERT pre-trained weights, which are built on large-scale domain corpora, and adversarial perturbations are applied to the embedding layer to improve the robustness of the model. Experimental results showed that the proposed model achieved the highest F1 scores (83.93% and 90.31%) on two corpora with large sample sizes, namely, AIMed and BioInfer, respectively, compared with the previous method. It also achieved comparable performance on three corpora with small sample sizes, namely, HPRD50, IEPA, and LLL.

Keywords: adversarial training, information extraction, natural language processing, pre-trained language model, protein-protein interaction

This research was supported by The National Key Research and Development Program of China, the China government [2016YFD0300710, Strategies and Technical Regulations for the Unified Prevention and Control of Major Diseases and Pests in the Main Grain Producing Areas].

http://doi.org/10.3837/tiis.2022.03.002 **ISSN : 1976-7277**

1. Introduction

Protein-protein interaction (PPI) is very important for understanding the molecular mechanism of biological processes [1]. The study of PPI has important reference significance for biomedical work, such as the study of drug targets [2] and analysis of signal proteins [3]. Abundant PPI information exists in biomedical literature in an unstructured form. Manual PPI extraction is costly and requires long time given the large number of published studies on PPI. Therefore, automatic PPI extraction from biomedical literature has become an important research field, which has attracted the attention of the majority of researchers.

In the previous work, extracting PPI based on co-occurrence and pattern matching is very popular [4-6]. However, in recent years, machine learning method with better performance has become the mainstream. Machine learning method constructs feature set based on feature engineering and kernel method, and then classified by support vector machine or other classifiers. The commonly used kernel functions are based on synchronous parse trees [7] and dependency parse trees [8]. Distributed smoothed tree kernel (DSTK) method proposed by [7] has achieved significant improvement compared with other kernel methods and machine learning methods. Some researchers also began to apply deep learning methods to PPI extraction and achieved satisfying performance given the wide application of deep learning technology in natural language processing. The first attempt to introduce deep learning into the field of PPI relation extraction, used feature engineering to construct feature sets and deep neural network (DNN) for modeling [9]. Although the first attempt to introduce deep learning technology to the PPI extraction problem used a deep learning model, it is still essentially a feature engineering-based method, which can be improved. Methods based on convolution neural network (CNN) further improve the accuracy of PPI extraction. They usually use word embedding as input features [10] and combine withshortest dependency path (SDP), lexical features, and semantic features. [11- 13]. The use of residual connection can increase the feature extraction ability of CNN, with the word embedding as input to obtain better results than machine learning methods. The experimental results show that deeper structures can directly learn sufficient features from the text itself [14]. Recurrent neural network (RNN)-based methods are also widely used [15-17]. The simple large capacity bidirectional long short term memory (LSTM) model achieves satisfactory results on large sample size datasets [15]. In addition, the transformer-based methods emphasize the integrity of global context representation, such as Bert [18] and BioBERT [19]. [20] added lexical features to the transformer structure and proposed an LBERT model based on lexical patterns to represent sentences. The accuracy of the LBERT model has been greatly improved compared with BioBERT. However, its performance is still far behind that of LSTM and CNN-based methods on large sample size datasets. Its fine-tuning methods from the official simple implementation are inapplicable to PPI extraction; thus the advantages of the large pre-trained language model cannot be fully expressed.

In general, the PPI extraction approach based on deep learning can be divided into three parts, namely, feature representation component, feature extraction component, and classifier component, as shown in **Fig. 1**.

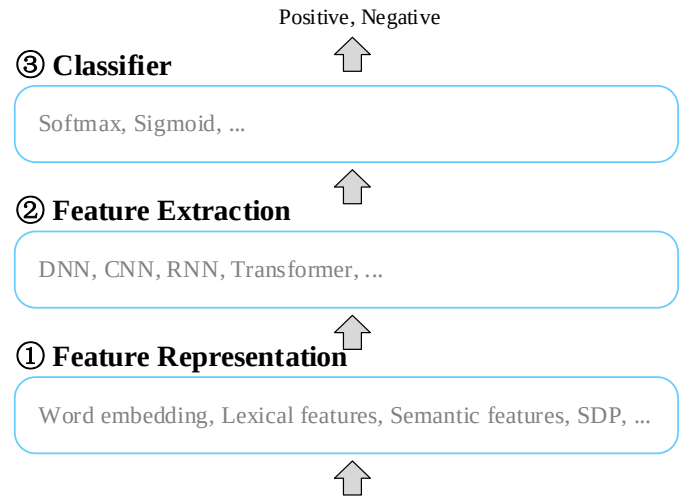


Fig. 1. Illustration of the PPI extraction approach based on deep learning

The feature representation component is used to convert text data into numerical data for subsequent parts of the model, which has a crucial impact on the performance of the model. Common feature representation components include word embedding, lexical features, semantic features, and SDP. The feature extraction component is used to model text, further abstract and extract text features, and generate vectors for final classification. Common feature extraction components include DNN, CNN, RNN, and transformer. In addition to feature extraction, a transformer can be used for feature representation. Relevant studies show that transformer structure is superior to LSTM, CNN, and other structures [21] However, in PPI extraction, the existing transformer-based model has not achieved significantly better performance than LSTM, CNN, and other traditional structures. The classifier is used to generate the final predicted label. A linear transformation layer and a sigmoid or softmax function are added to the top of the model to generate conditional probabilities in the category space. For two classification problems, using the sigmoid and the softmax function is roughly equivalent.

Moreover, to increase the generalization of the PPI extraction model, the protein named entity in the text is usually replaced with the common entity name, as follows: “PROTEIN1,” “PROTEIN2” or “PROTEIN,” where “PROTEIN1” and “PROTEIN2” are the target pairs, “PROTEIN2” stands for other protein named entities. For example, the sentence “*thymocyte activation induces the association of phosphatidylinositol 3-kinase and pp120 with CD5*.” contains three protein named entities,namely, “phosphatidylinositol 3-kinase,” “pp120,” and “CD5.,” When considering the interaction between “phosphatidylinositol 3-kinase” and “pp120,” the sentence is rewritten as follows: “*thymocyte activation induces the association of PROTEIN1 and PROTEIN2 with PROTEIN*.,” When we pay attention to the interaction between “pp120” and “CD5,” it is rewritten as “*thymocyte activation induces the association of PROTEIN and PROTEIN1 with PROTEIN2*.” It leads to many samples with small differences in the text data, seriously affecting the robustness of the PPI extraction model.

To address these problems, based on the BERT architecture, the adversarial training method is proposed, and the ADVBERT model and its two variant structures are proposed. In the training process, the token embedding input is perturbed to improve the quality of the learned word embedding and the overfitting problem. The main contributions of this study are as follows:

(1) The limitation of the previous work is that the semantic and syntactic information captured is insufficient. This work fully uses the semantic and syntactic information of the large pre-trained language model, combined with the pre-training weights obtained from a lot of text in the field. In addition, the BERT architecture based on the small neural network is improved, and a new state-of-the-art method on PPI extraction is achieved.

(2) The previous work should be improved in terms of robustness. This work introduces adversarial training into the PPI extraction approach and proposes an adversarial training method for PPI-related texts. The PPI extraction model obtains higher robustness in the presence of many small difference samples by applying the adversarial perturbations to the weight matrix of token embedding.

2. Related work

2.1 Pre-trained language model

In the field of natural language processing, a deep network structure is trained using a large-scale unlabeled text corpus to obtain a set of model parameters. This deep network structure is usually called the pre-trained model. The pre-trained model can use general information in the large-scale unlabeled text to improve the effectiveness of subsequent specific tasks and reduce the requirement for the amount of labeled data; it can also handle some scenes, where obtaining a large amount of labeled data is difficult [22].

Most early pre-trained models used static techniques, such as Word2Vec [23] and GloVe [24]. These methods can learn shallow representations of text, which can improve the downstream tasks. However, static pre-training technology cannot solve the problem of polysemous words that often appear in natural language. In response to this problem, researchers began to apply dynamic pre-training technology to pre-trained language models. ELMo [25] uses two-layer two-way LSTM with residuals to train the language model, captures contextual information, and solves the problem of ambiguity. [26] proposed a multistage migration method and fine-tuning the pre-training model skills to provide important guidance for the subsequent development of pre-training technology. Transformer [21] usually has better results than LSTM, and has achieved effective results on tasks such as machine translation; it has been applied to pre-trained language models. The generative pre-training (GPT) model [27] stacks 12 transformer substructures, which have sufficient characterization capabilities and significantly improve the effect of downstream tasks. The GPT model is still essentially a one-way language model, and its ability to model semantic information is limited. To solve this problem, the BERT model [18] achieved the effect of a two-way language model by randomly covering part of the words in the input text sequence during pre-training. In addition, [28] proposed a model based on federated learning. The emergence of the BERT model has greatly promoted the development of the field of natural language processing and reached a new state-of-the-art in several natural language processing tasks. However, the BERT pre-trained weights trained on the general corpus cannot achieve satisfactory results in some cases due to the more complex textual sentence patterns in the biomedical field related to PPI and more professional terms. BioBERT [19] studied how to apply the pre-training

language model BERT to the field of biomedicine. Through the almost identical architecture on the task, after pre-training on the biomedical corpus, BioBERT is used in many biomedical text mining tasks. It is much better than BERT and the previous SoTA models. However, [20] applied BioBERT's official code to PPI relation extraction and failed to achieve good results; the potential of BioBER's pre-training weights should be further developed.

2.2 Adversarial training methods

Adversarial training was first applied in the field of computer vision, and adversarial samples generated based on adversarial disturbances were added to the training process to solve the security problem of deep learning models. In recent years, with the development of deep learning technology in the field of natural language processing, adversarial training based on fighting perturbations has also been used in the field of natural language processing.

[29] found that after adding small perturbations that are invisible to the naked eye, the deep learning model provides false predictions with high confidence. Thus, the adversarial sample attracts the attention of the majority of researchers. [30] believed that this impact is caused by the linear behavior of deep neural networks in high dimensions given that small perturbations on the input may greatly affect the deep learning model. In addition, an adversarial objective function based on fast gradient sign method (FGSM), which enables adversarial training to be carried out at the same time as model training, was proposed. Experiments have shown that this regulation method is effective and can improve the robustness of deep learning models to small perturbations. [31] applied adversarial training to text problems, based on FGSM, removed the sign function, used the L2 norm to scale the gradient, and proposed the fast gradient method (FGM). Different from image data, text data are mostly in discrete form, and directly adding adversarial perturbations to the raw data is impossible. Therefore, FGM adds small perturbations to the embedding layer. Although the embedding layer with disturbance cannot correspond to the actual text input, this method can still effectively improve the performance of the model. [32] analyzed the adversarial training and summarized it as a minimum–maximization problem: the disturbance that maximizes the loss of the original model is found,as well as the most robust parameters of the model when the data error rate is the largest.

To the best of our knowledge, use of adversarial training to improve the effect of PPI extraction has not been explored. Many samples with small differences are formed because the protein named entity is replaced with the general entity name in the PPI extraction research, thereby affecting the robustness of the PPI model. Adversarial training can improve this limitation to a certain extent.

3. Method

The basic architecture of the adversarial bidirectional encoder representations from transformers (ADVBERT) model includes input embeddings, BERT architecture, and classifier, as shown in **Fig. 2**.

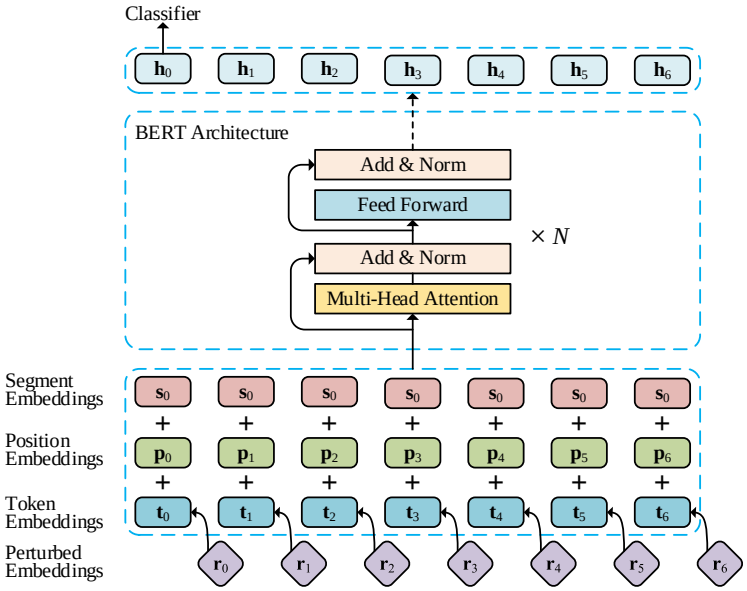


Fig. 2. Basic architecture of the ADVBERT model

3.1 Input embeddings

3.1.1 Tokenization

This work uses wordpiece [33] for word segmentation to reduce the size of the vocabulary. The definition of the word is separated from the tense voice by breaking the word into smaller units. This method can obtain streamlined and richer vocabulary. After the word segmentation, a special token [CLS] for classification is added to the beginning of the sequence, and a special token [SEP] for identifying the end of a sentence is added to the end to satisfy the requirements of using pre-trained weights. For example, the word sequence of "Levels of PROTEIN1 were slightly increased following PROTEIN2 treatment" after tokenization is as follows: *"[CLS]," "Level," "##s," "of," "PR," "##OT," "##EI," "##N," "##1," "were," "slightly," "increased," "following," "PR," "##OT," "##EI," "##N," "##2," "treatment,"*and *"[SEP]."* The word sequence is converted into the corresponding integer id sequence as the input of the token embeddings.

3.1.2 Input embeddings

Segment embeddings are used to represent the sentence. For PPI extraction, only one segment is required; it is uniformly defined as $\boldsymbol{s}_0$. Position embeddings are used to encode position information. This work uses sine and cosine functions to construct position embeddings. The length of the input sequence is n, $i \in [0, n)$ represent the i-th position in the sequence, and the embedding layer dimension is d_e.Then, $\boldsymbol{s}_0 \in R^{d_e}$, $\boldsymbol{t}_i \in R^{d_e}$, and the position embeddings can be computed as follows:

$$\boldsymbol{p}_i^{(2k)} = \sin\left(i/10000^{2k/d_e}\right) \quad (1)$$

$$\boldsymbol{p}_i^{(2k+1)} = \cos\left(i/10000^{2k/d_e}\right) \quad (2)$$

where $k = d_e/2$, $\boldsymbol{p}_i^{(2k)}$ represents the value of $\boldsymbol{p}_i$ in the $2k$ dimension.

Token embeddings is used to represent word embedding vectors. Adversarial perturbations are applied into token embeddings to increase the robustness and effectively regulate the

model. The FGM method directly adds perturbations to the embeddings [31], but this method requires disassembly and reconstruction of the model; it is diffcult to implement. Perturbing the embedding parameter matrix can also play a regularizing role because the output of embeddings is calculated from the embedding parameter matrix. Adversarial perturbations are added to the parameter matrix of token embeddings. Perturbation values are calculated by the gradient of the parameter matrix of token embeddings, and the L2 norm of the parameter matrix and an adjustable adversarial coefficient are used to scale the perturbation values. The parameter matrix of token embeddings is set to be $\boldsymbol{W}_T$, as follows:

$$\boldsymbol{R}_{adv} = [\boldsymbol{r}_0, \boldsymbol{r}_1, \cdots \boldsymbol{r}_{n-1}] = \alpha \frac{\nabla_{\boldsymbol{W}_T} L(\boldsymbol{\theta}, \boldsymbol{x}, y)}{\|\nabla_{\boldsymbol{W}_T} L(\boldsymbol{\theta}, \boldsymbol{x}, y)\|_2} \tag{3}$$

where α is the adversarial coefficient, L is the loss function, $\boldsymbol{\theta}$ is the model parameter, and $\boldsymbol{x}, y$ are the input and output of the model, respectively. During training, $\boldsymbol{W}_T = \boldsymbol{W}_T + \boldsymbol{R}_{adv}$, and the perturbations are removed after parameter update, $\boldsymbol{W}_T = \boldsymbol{W}_T - \boldsymbol{R}_{adv}$. This type of perturbation is different from random noise. The randomly generated noiseunnecessarily increases the gradient, but the perturbations generated in this approach always increase the gradient. Thus, the regularization effect is stronger.

The output of the entire input embeddings $\boldsymbol{E}$ is the sum of the three embeddings obtained as follows:

$$\boldsymbol{e}_i = \boldsymbol{s}_0 + \boldsymbol{t}_i + \boldsymbol{p}_i \tag{4}$$

$$\boldsymbol{E} = [\boldsymbol{e}_0, \boldsymbol{e}_1, \cdots \boldsymbol{e}_{n-1}] \in R^{n \times d_e} \tag{5}$$

3.2 BERT Architecture

3.2.1 Transformer blocks

Attention mechanism has not only been widely used in the field of computer vision [34-37], but also in the field of natural language processing. Attention-based BERT architecture is composed of N transformer blocks. Transformer blocks mainly include two parts, namely, Multi-Head attention and Feed Forward. The attention calculation method is Scaled Dot-Product Attention [18], as follows:

$$ATT(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = softmax\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V} \tag{6}$$

where d_k is the dimension of $\boldsymbol{K}$.

Multi-Head attention allows the model to focus on the letters of different representation subspaces from different positions. The calculation method of a single attention head adopts self-attention calculation. $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ are obtained by a linear function of the output of input embeddings $\boldsymbol{E}$, and Multi-Head attention is obtained by concatenating the attention output of each head, as follows:

$$head_i = ATT\left(\boldsymbol{E}\boldsymbol{W}_i^Q, \boldsymbol{E}\boldsymbol{W}_i^K, \boldsymbol{E}\boldsymbol{W}_i^V\right), i \in [1, n_H] \tag{7}$$

$$\boldsymbol{M} = concat(head_1, \cdots, head_H)\boldsymbol{W}_M \in R^{n \times d_e} \tag{8}$$

where n_H is the number of attention heads, $\boldsymbol{W}_i^Q, \boldsymbol{W}_i^K, \boldsymbol{W}_i^V \in R^{d_e \times d_w}$, $d_w = d_e/n_H$, $\boldsymbol{W}_M \in R^{d_e \times d_e}$. When the output of the Multi-Head attention is connected with the input residually, the layer normalization [33] is applied, as follows:

$$\boldsymbol{E} = LayerNorm(\boldsymbol{E} + \boldsymbol{M}) \in R^{n \times d_e} \tag{9}$$

where $LayerNorm$ is layer normalization.

The Feed Forward operation consists of two linear transformations, with a GELU activation function in between; it also performs residual connection and layer normalization operations, as follows:

$$\boldsymbol{F} = GELU(\boldsymbol{E}\boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2 \in R^{n\times d_e} \quad (10)$$

$$\boldsymbol{E} = LayerNorm(\boldsymbol{E} + \boldsymbol{F}) \in R^{n\times d_e} \quad (11)$$

where $\boldsymbol{W}_1 \in R^{d_e\times d_1}, \boldsymbol{W}_2 \in R^{d_1\times d_e}$, and d_1 is the intermediate size.

This operation is repeated N times to obtain the final hidden layer vectors, $\boldsymbol{H} = [\boldsymbol{h}_0, \boldsymbol{h}_1, \cdots \boldsymbol{h}_{n-1}] \in R^{n\times d_e}$, and the $\boldsymbol{h}_0$ corresponding to the classification identifier [CLS] is input to the classifier for classification.

3.2.2 Pre-trained weights

The BERT architecture uses a bidirectional transformer for encoding and a masked language model during pre-training, a small number of words are replaced with MASK or another random word with a small probability during training, thereby enhancing the context memory. The PPI-related text is very closely related to the context, and fully integrating the context is necessary to obtain sufficient semantic feature representation. Two open-source pre-trained weights are used for comparative research to use the information in the large-scale unlabeled corpus. One is the *BERT-Base cased* released by Google Research (https://github.com/google-research/bert), and the other is *BioBERT-Base v1.1* (+PubMed 1M) released by DMIS LAB (https://github.com/dmis-lab/biobert). The capacity, structure, configuration, and vocabulary of the two pre-trained weights are the same, as shown in **Table 1**.

Table 1. Parameter description of BERT architecture

Parameter	Description	Value
d_e	dimension of embeddings and hidden layer	768
N	number of transformer blocks	12
n_H	number of attention heads	12
d_1	intermediate size	3072

3.2.3 Fine-tuning

The self-attention mechanism in the transformer allows BERT architecture to fine-tune specific downstream tasks by exchanging appropriate inputs and outputs. The knowledge obtained during pre-training on large-scale unlabeled corpus can be transferred to specific tasks during fine-tuning. PPI extraction relies on domain knowledge and a large number of semantic and syntactic features. Fine-tuning on large-scale pre-training language models can greatly improve the efficiency of PPI extraction. We consider the sentence after replacing the named entity as input, and whether PPI exists between the two proteins of interest as the binary classification output.

3.3 LSTM-based variant

In the original architecture of BERT, only the hidden layer vector corresponding to [CLS] is used, losing part of the text information. We design a variant structure based on LSTM, named ADVBERT-LSTM, as shown in **Fig. 3** to fully use of information in addition to [CLS], considering the effectiveness of bidirectional LSTM in PPI extraction [15].

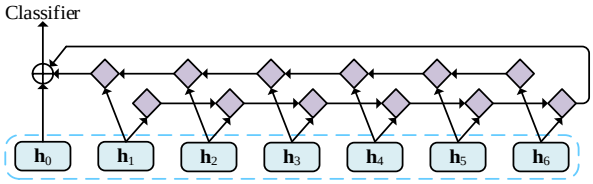


Fig. 3. LSTM based variant structure

The hidden layer output vectors except $\boldsymbol{h}_0$ are processed by bidirectional LSTM and concatenated with $\boldsymbol{h}_0$ to generate the final vector for classification, as follows:

$$\overrightarrow{\boldsymbol{h}_l} = LSTM(\boldsymbol{h}_1 \rightarrow \boldsymbol{h}_{n-1}) \in R^{d_l} \tag{12}$$

$$\overleftarrow{\boldsymbol{h}_l} = LSTM(\boldsymbol{h}_{n-1} \rightarrow \boldsymbol{h}_1) \in R^{d_l} \tag{13}$$

$$\boldsymbol{h}_0 = concatenate\left(\boldsymbol{h}_0, \overrightarrow{\boldsymbol{h}_l}, \overleftarrow{\boldsymbol{h}_l}\right) \in R^{d_e+2\times d_l} \tag{14}$$

where d_l is the dimension of the LSTM hidden units.

The LSTM structure can model the text sequence satisfactorily and capture the long-distance dependence in the text. The gating mechanism used in LSTM can effectively improve the vanishing gradient problem. The combination of LSTM and BERT can further improve the performance of PPI extraction.

3.4 CNN-based variant

The hidden layer output vectors obtained by the BERT architecture have a higher level of abstraction and contain sufficient position information. Therefore, we design another variant structure based on CNN to effectively capture other information besides [CLS], named ADVBERT-CNN, as shown in **Fig. 4**.

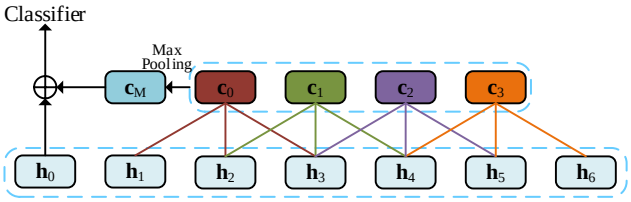


Fig. 4. CNN based variant structure

A filter with a window size of 3 is used to perform convolution operations on the hidden layer output vectors except $\boldsymbol{h}_0$, as follows:

$$\boldsymbol{C} = CNN(\boldsymbol{h}_{1:3}, \boldsymbol{h}_{2:4}, \cdots, \boldsymbol{h}_{n-3:n-1},) \in R^{(n-3)\times d_c} \tag{15}$$

where d_c is the dimension of the CNN hidden units. Then the features are further extracted through global max pooling and then concatenated with $\boldsymbol{h}_0$ to generate the final vector for classification, as follows:

$$\boldsymbol{C_M} = GlobalMaxPooling(\boldsymbol{C}) \in R^{d_c} \tag{16}$$

$$\boldsymbol{h}_0 = concat(\boldsymbol{h}_0, \boldsymbol{C_M}) \in R^{d_e+d_c} \tag{17}$$

CNN structure can identify indicative substructures in the input data and capture the local features that contribute the most to the prediction task. BERT architecture generates features that contain rich semantic and word meaning information through several transformer blocks. The CNN structure further combines these features through convolution and pooling operations to better represent the entire text sequence.

3.5 Classifier

A linear transformation layer and a softmax function are used to $\boldsymbol{h}_0$, which is the final form for classification, to generate conditional probabilities in the category space, as follows:

$$\mathbf{p} = softmax(\boldsymbol{h}_0 \boldsymbol{W}_c + \boldsymbol{b}_c) \in R^2 \quad (18)$$

where $\boldsymbol{W}_c \in R^{d_e \times 2}$. Then, the predicted label $\hat{y}$ is outputted through the argmax function:

$$\hat{y} = argmax(\boldsymbol{p}) \quad (19)$$

The loss function uses cross-entropy:

$$loss = -ylnp_1 - (1-y)\, ln(1-p_0) \quad (20)$$

The optimization goal of the model is to minimize the loss value of all training samples.

4. Results

4.1 Datasets

Various institutions provide a series of standard PPI extraction corpora to facilitate the majority of researchers to study the problem of PPI relation extraction. The more widely used approach in the existing research are AIMed [38], BioInfer [39], HPRD50 [40], IEPA [41], and LLL [42]. Each corpus contains multiple sentences from the biomedical literature, and each sentence contains one or more protein pairs. The protein named entity has also been annotated. Each protein pair is a sample;whether each protein pair has a mutual relationship, the interaction relationship is annotated by experts in the relevant field. The protein pairs that interact with each other are positive samples, and the nonexistent protein pairs are negative samples.

Some annotation differences in the AIMed and BioInfer corpora are processed following the principle of effective length. After the annotation is completed, samples that do not contain PROTEIN1 and PROTEIN2 at the same time are deleted. After processing, the statistics of each corpus are shown in **Table 2**.

Table 2. The statistics of PPI extraction corpora

Corpus	Number of samples	Positive samples	Negative samples	Ratio
AIMed	5669	995	4674	0.21
BioInfer	10090	2425	7665	0.32
HPRD50	433	163	270	0.60
IEPA	817	335	482	0.70
LLL	330	164	166	0.99

4.2 Evaluation metrics

We use standard evaluation indicators, including precision (P), recall (R), and F1 score (F1) as metrics to evaluate the model performance. The calculation method of each metric is as follows:

$$P = \frac{TP}{TP+FP} \times 100\% \quad (21)$$

$$R = \frac{TP}{TP+FN} \times 100\% \quad (22)$$

$$F1 = \frac{2PR}{P+R} \times 100\% \quad (23)$$

where TP is the number of samples that are positive and classified as positive, FN is the number of samples that are positive and classified as negative, FP is the number of samples that are negative and classified as positive, and TN is the number of samples that are negative and classified as negative.

4.3 Experimental settings

Since these corpora do not officially divide the training and test sets, in this work, 10-fold cross-validation is used; it is the most reliable evaluation method. Some hyperparameter settings are shown in **Table 3**.

Table 3. Hyperparameter settings

Hyperparameter	Description	Value
epochs	epochs of training	10
batch size	batch size of training	8
lr	Learning rate	2e-5
n	maximum length of the input sequence	128
d_l	dimension of LSTM hidden units	256
d_c	dimension of CNN hidden units	128

We trained our model on an NVIDIA GeForce GTX 1660 GPU with the Adam optimizers with $\beta_1 = 0.9$, $\beta_1 = 0.98$, and $\varepsilon = 10^{-9}$. We used open-source libraries TensorFlow (https://www.tensorflow.org/), bert4keras (https://github.com/bojone/bert4keras), and Keras (https://keras.io/) to implement our model under the Windows10 environment. The model is directly used for prediction after 10 epochs of training in the cross-validation of each fold.

4.4 Experimental results

The proposed model is compared with state-of-the-art approaches in the traditional machine learning methods DSTK, and other deep learning methods including DNN [9], MCCNN [10], sdpCNN [11], McDepCNN [12], LSTM [15], tLSTM [16], DEEPCNN [14], LBERT [20]. The compared methods are briefly introduced as follows. The comparison results are shown in **Table 4**. The results of the ADVBERT-CNN and ADVBERT-LSTM models are obtained using the BioBERT pre-trained weights and the adversarial coefficient α =0.25, and the standard deviation of the 10-fold cross-validation is in parentheses.

Table 4. Comparison results (%)

Model	AIMed			BioInfer			HPRD50			IEPA			LLL		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
DSTK	68.91	73.24	71.01	75.70	76.90	76.29	76.25	84.15	80.0	75.85	85.15	80.23	87.31	91.18	**89.20**
DNN	51.51	63.38	56.12	53.89	72.9	61.63	66.95	83.98	74.23	58.72	**92.37**	71.28	75.84	**91.81**	82.00
MCCNN	76.41	69.00	72.45	81.30	78.10	79.62	-	-	-	-	-	-	-	-	-
sdpCNN	64.8	67.8	66.0	73.4	77.0	75.2	-	-	-	-	-	-	-	-	-
McDepCNN	67.3	60.1	63.5	62.7	68.2	65.3	-	-	-	-	-	-	-	-	-
LSTM	78.8	75.2	76.9	87.0	87.4	87.2	-	-	-	-	-	-	-	-	-
tLSTM	81.4	81.9	81.6	88.9	89.3	89.1	81.7	82.3	81.3	78.6	78.7	78.5	84.8	84.3	84.2
DEEPCNN	79.0	76.8	77.6	87.4	86.5	86.9	74.9	82.8	77.7	71.6	80.6	75.5	80.5	87.2	83.2

LBERT	73.6	74.5	74.0	73.0	72.5	72.8	**85.8**	85.2	**85.5**	81.4	86.2	83.7	83.8	88.4	86.0
ADVBERT-LSTM	**84.74** (4.12)	**83.39** (2.49)	**83.93** (1.19)	89.94 (2.43)	**90.08** (2.43)	89.97 (1.25)	83.66 (8.99)	**87.17** (8.81)	84.78 (5.61)	**85.34** (6.37)	84.15 (5.75)	84.48 (3.77)	**89.62** (9.26)	88.31 (8.61)	88.65 (7.45)
ADVBERT-CNN	84.34 (3.89)	81.40 (3.53)	82.72 (1.93)	**91.37** (1.65)	89.28 (2.30)	**90.31** (1.83)	83.40 (6.44)	85.40 (8.46)	83.94 (4.12)	84.13 (4.83)	85.95 (7.02)	**84.88** (4.80)	87.98 (8.81)	89.60 (6.22)	88.61 (6.56)

The proposed model has achieved the highest F1 scores on the three corpora of AIMed, BioInfer, and IEPA. The comparison between ADVBERT-LSTM and the previous state-of-the-art method indicates that, compared with tLSTM, the F1 score of the AIMed increased by 2.3 percentage points, 0.8 percentage points for the BioInfer corpus.The F1 score of the HPRD50 was 0.8 percentage points lower than that of LBERT, and that of the IEPA increased by 0.7 percentage points. The F1 score of the LLL was 0.55 percentage points lower than DSTK. DSTK and LBERT performed poorly on the two large sample size corpora of AIMed and BioInfer. However, the proposed model has achieved significant improvement on the AIMed and BioInfer corpora compared with the two models. For the three small sample size corpora of HPRD50, IEPA, and LLL, the proposed model has also achieved a significant improvement over tLSTM. The proposed model can achieve better performance on the large sample size and small sample size corpora.

The use of large pre-trained language models can greatly improve the performance of downstream tasks, given that the lexical and syntactic features required by the downstream tasks are learned from a large amount of corpus, especially the corpus in the field, in the pre-training process. Fine-tuning with the target corpus can complete the PPI extraction task effectively, and can improve the problem of the deep learning model being restricted by the sample size to a certain extent, because the features that cannot be fully learned on the small sample size data may have been effectively represented in the pre-training process.

5. Discussion

5.1 Comparisons between different methods

The advantage of the deep learning models over the traditional machine learning models is that manually designing feature engineering is unnecessary. The model itself can mine the potential features of the data, as much as possible, but it requires a sufficient amount of data to be effective. Therefore, on the two small-sample datasets of IEPA and LLL, the DSTK method using kernel function and machine learning model has certain advantages over the two deep learning methods of tLSTM and DEEPCNN. However, for AIMed and BioInfer datasets with large sample sizes, the advantages of deep learning methods are highlighted. Deep learning methods, such as LSTM, tLSTM, and DEEPCNN, have achieved significant improvements compared with DSTK methods. The LBERT method that also uses a large-scale pre-training language model is based on deep learning technology. However, it has learned sufficient lexical and syntactic features from a large amount of unlabeled corpus during the pre-training process. Thus, it also obtained considerable performance on small sample size datasets.

The proposed ADVBERT–CNN and ADVBERT–LSTM models use adversarial training to further exert the advantages of large-scale pre-training language models and enhance the robustness of the models. Through the combination with CNN and LSTM structure, the ability to mine the potential features of text data is improved, and the accuracy of text feature modeling is enhanced. Therefore, the proposed model has achieved good performance on large sample size and small sample size data. In particular, the highest F1 scores were achieved on

the three corpora, namely, AIMed, BioInfer, and IEPA. On the two other datasets, the F1 scores obtained are also very close to the best performance methods.

In summary, the proposed method further improves the performance of PPI extraction, and at the same time introduces large-scale pre-training language models and adversarial training techniques into the field of PPI extraction.

5.2 Performance with different pre-trained weights

Experiments were performed using the general BERT (*BERT-Base cased*) pre-trained weights and the domain-related BioBERT (*BioBERT-Base cased v1.1*) pre-trained weights to explore the impact of different pre-trained weights. The experimental results of the original ADVBERT model are shown in **Fig. 5**.

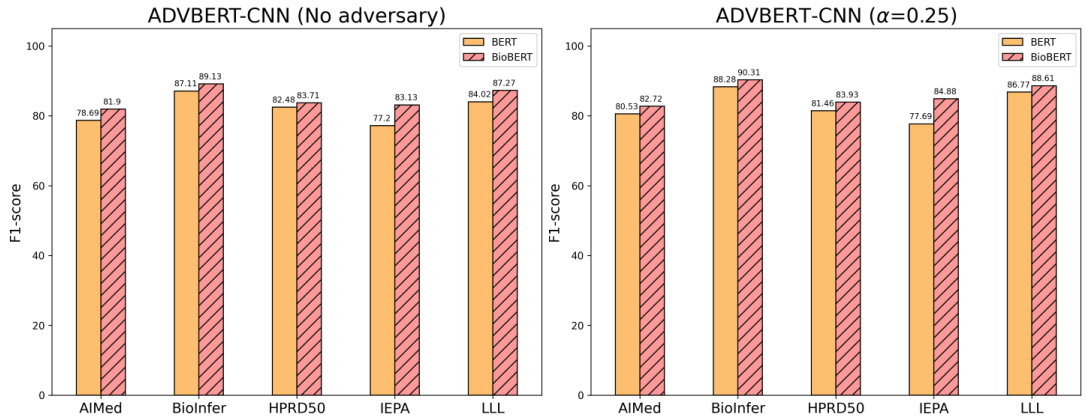


Fig. 5. F1-score (%) of the original ADVBERT model using BERT and BioBERT pre-training weights with adversarial training (a) and without adversarial training (b)

The experimental results of the LSTM based variant ADVBERT-LSTM model are shown in **Fig. 6**.

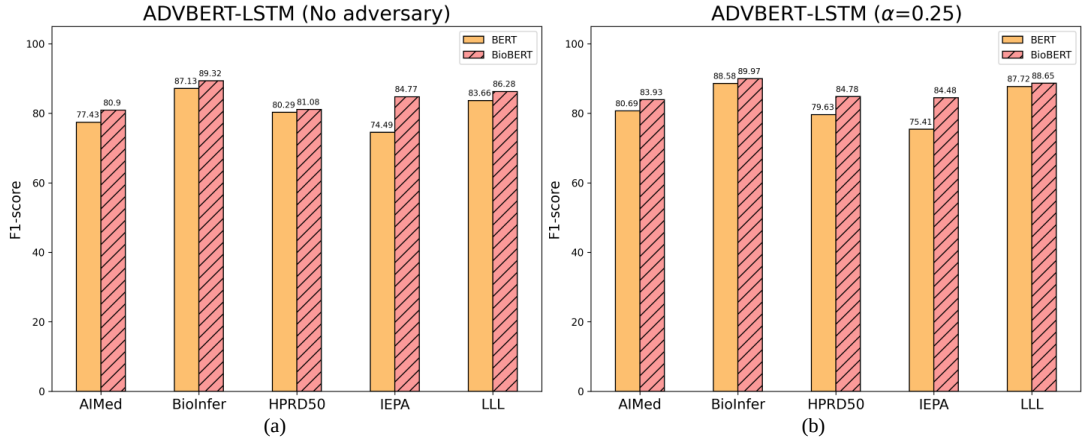


Fig. 6. F1-score (%) of ADVBERT-LSTM model using BERT and BioBERT pre-training weights with adversarial training (a) and without adversarial training (b)

The experimental results of the CNN based variant ADVBERT-CNN model are shown in **Fig. 7**.

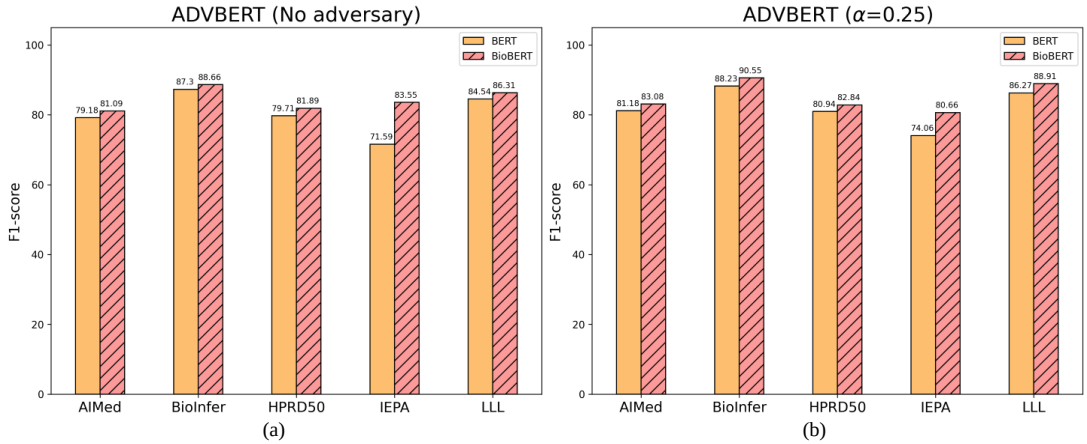


Fig. 7. F1 score (%) of the ADVBERT-CNN model using BERT and BioBERT pre-training weights with adversarial training (a) and without adversarial training (b)

Regardless of the variant structure used and whether adversarial training is applied, the results obtained by BioBERT pre-trained weights are better than those obtained by BERT pre-trained weights. This finding shows that pre-training with corpus in the domain can help improve the effect of downstream tasks. On the large sample size corpora of AIMed and BioInfer, compared with the BERT pre-trained weight, using the BioBERT pre-trained weight can usually achieve a 1 to 2 percentage point increase in F1 score. On the HPRD50, IEPA, and LLL with smaller sample sizes, the gap is even more evident. The sample size limits the effect of the deep learning model given that the large-capacity model cannot be fully trained with a small amount of data. However, the BioBERT pre-trained weights are constructed on abundant text in the field. Thus, it contains more domain-related semantic and syntactic information, which can be well transformed to downstream tasks.

5.3 Performance using different variants

Experiments were carried out using the original ADVBERT model and the two variants based on CNN and LSTM (using BioBERT pre-trained weights). The results are shown in **Fig. 8**.

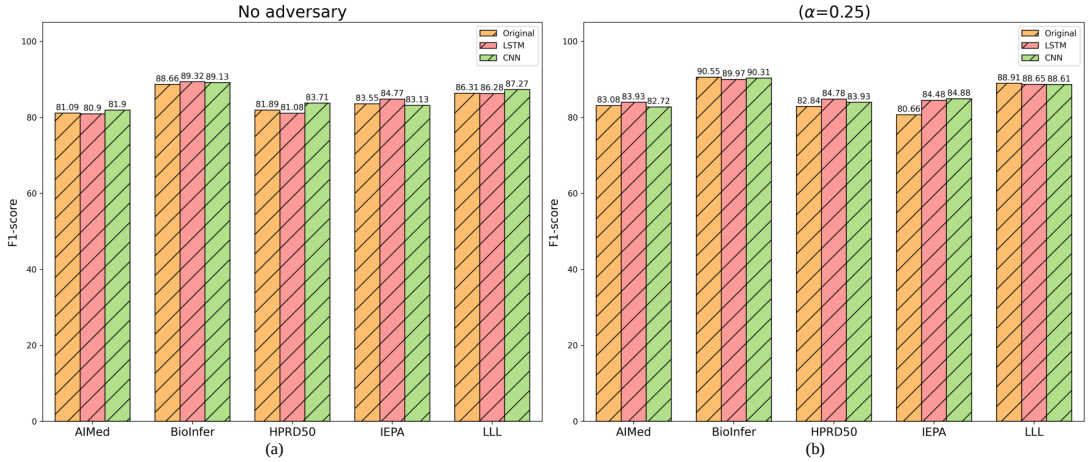


Fig. 8. F1-score (%) of the original ADVBERT model and its variants with adversarial training (a) and without adversarial training (b)

Fig. 8 shows that regardless of whether adversarial training is used or not, the performance of original ADVBERT and the two variants on AIMed and BioInfer is unremarkably different. When adversarial training is unused, the LSTM based variant achieves the best performance on the IEPA. However, the CNN-based variant achieves the best performance on the HPRD500 and LLL. When using adversarial training (α =0.25), the LSTM-based variant achieved the best performance on the HPRD500 and LLL, and the CNN-based variant achieved the best performance on the IEPA dataset. In summary, the use of variant structures can further utilize hidden layer information in addition to [CLS], and can slightly improve the accuracy of PPI extraction.

5.4 Effect of the adversarial coefficient

Experiments were carried out on the original ADVBERT model and its variants with different adversarial coefficient values and without adversarial training (using BioBERT pre-trained weights), to illustrate the advantages of adversarial training and the effect of adversarial coefficient α.

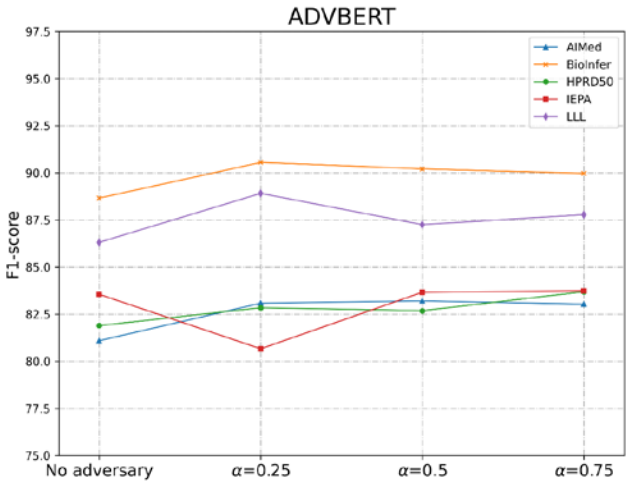


Fig. 9. Effect of different adversarial coefficients on the original ADVBERT model

Fig. 9 shows the effect of different adversarial coefficients on the original ADVBERT model. **Fig. 9** shows that all corpora, except the IEPA, achieved the highest F1 scores when α =0.25. When α exceeds 0.25, as α increases, F1-scores have declined to a certain extent, but overall it is higher than that without adversarial training.

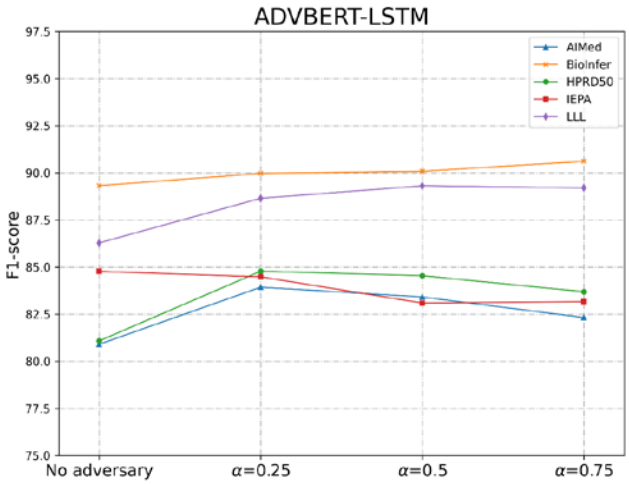


Fig. 10. Effect of different adversarial coefficients on ADVBERT-LSTM model

Fig. 10 shows the effect of different adversarial coefficients on LSTM based variant ADVBERT-LSTM. On the IEPA, ADVBERT-LSTM obtained the highest F1-score without using adversarial training. The performance of using adversarial training on other corpora is still better than not using the approach. On BioInfer and LLL, the F1 score increases with the increase in α. However, the F1 score still shows a downward trend after α exceeds 0.25 on AIMed and HPRD50.

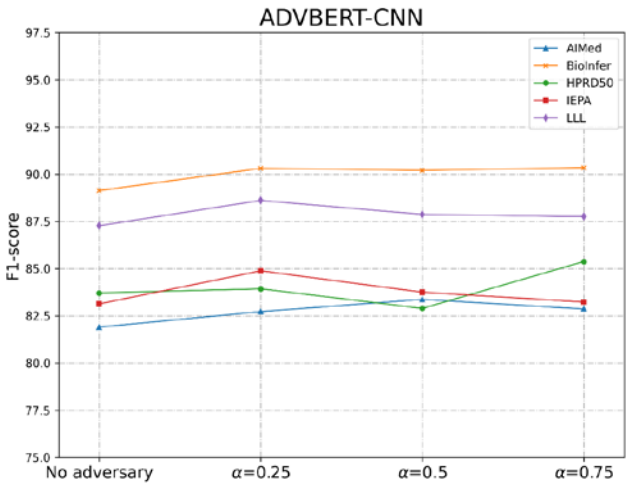


Fig. 11. Effect of different adversarial coefficients on ADVBERT-CNN model

Fig. 11 shows the effect of different adversarial coefficients on CNN-based variant. For ADVBERT-CNN model, the performance of using adversarial training is better than that of not using it. In most corpora, the F1-score still shows a downward trend after α exceeds 0.25.

In summary, the performance of using adversarial training is better than not using adversarial training in most cases. Better performance can be achieved when α is 0.25, but extremely large α may also cause a decrease in performance.

5.4 Error analysis

In this section, several typical error cases are selected for analysis from the prediction results of each corpus, which mainly includes three types, namely, protein named entity nesting, the sentence contains too many named entities of protein, and the annotation of ambiguity.

The largest proportion is the error caused by the protein named entity nesting, such as "*A bacterially expressed 318-amino acid fragment, PROTEIN1 (418-736), containing the amphipathic helix region, was able to bind PROTEIN2 alpha.* " In this sentence, *PROTEIN1* refers to "Ht 31" and *PROTEIN2* refers to "RII", "Ht 31" interacts with "RII alpha" but not with "RII," the model inaccurately predicts it to be positive.

Sentences containing many named entities of protein, mostly in BioInfer. For instance, "*Abundance of PROTEIN , PROTEIN , PROTEIN and PROTEIN , PROTEIN1 PROTEIN , and PROTEIN is not affected by the differential PROTEIN2 expression.* " A large number of named entities of protein leads to intricate interactions that should be predicted. These named entities of protein form many pairs, but the textual information of sentences is very limited. The model can accurately predicts most pairs, but it also inaccurately predicts on a small number of pairs.

The annotation of ambiguity, such as "*Transient expression of PROTEIN and a null mutant of PROTEIN1 (PTP-1CM) in COS cells resulted in an increase in tyrosyl phosphorylation of PROTEIN2 and its interaction with PTP-1CM*" in HPRD50, in which "CD22" appeared twice. For PTP-1C and the previous CD22, the label given by the corpus is false. However, for PTP-1C and the latter CD22, the label given by the corpus is true, thereby causing interference to the model.

5.5 Cross-corpus evaluation

Cross-corpus evaluation indicates training on one corpus and testing on another corpus to evaluate the generalization of the PPI extraction model. We only use AIMed and BioInfer as the training corpora because the sample size of other corpora is extremely small. The experimental results using AIMed as the training corpus are shown in **Table 5**. The experimental results with BioInfer as the training corpus are shown in **Table 6**. PIPE [43] is a knowledge-based state-of-the-art method for PPI extraction cross-corpus evaluation.

Table 5. Cross-corpus evaluation results using AIMed as the training corpus (F1-score %)

Model	BioInfer	HPRD50	IEPA	LLL
PIPE	**58.2**	69.4	**69.0**	**75.2**
McDepCnn	48.0	-	-	-
LSTM	49.3	-	-	-
tLSTM	45.0	39.1	37.9	33.5
ADVBERT-LSTM	43.40	**74.81**	26.02	37.86
ADVBERT-CNN	41.29	69.37	30.85	26.45

Table 5 shows that when AIMed is used as the training corpus, the proposed model performs poorly on most corpora. The training models on AIMed become over-fitting when learning various complex situations because the data in AIMed are the most complex. However, knowledge-based methods can avoid this limitation.

Table 6. Cross-corpus evaluation results using BioInfer as the training corpus

Model	AIMed	HPRD50	IEPA	LLL
PIPE	52.1	71.3	72.3	78.5
McDepCnn	49.9	-	-	-
LSTM	50.7	-	-	-
tLSTM	50.0	45.5	40.0	33.5
ADVBERT-LSTM	**54.55**	**78.93**	75.89	78.05
ADVBERT-CNN	54.20	77.69	**78.21**	**79.88**

Table 6 shows that when BioInfer was used as the training corpus, the proposed model achieved the highest F1-scores in the remaining four corpora. The BioInfer corpus has the largest amount of data without interference from a large number of complex cases; thus the model performs better in cross-corpus evaluation.

6. Conclusion

The ADVBERT model and its two variant structures are proposed for the PPI extraction task. The BioBERT pre-trained weights constructed on the large-scale domain corpus are used to enhance the learning of the semantic features and syntactic features of the PPI-related text. Adversarial training is introduced, and adversarial perturbations are applied to the token embedding of the model, thereby improving the robustness of the PPI extraction model. Experiments on five PPI extracting public corpora show that the proposed approach has achieved significant improvement. In the future, we will continue to explore how to further utilize the advantages of the large pre-trained language models, and find more effective approaches to generate adversarial perturbations to improve the generalization of the PPI extraction model.

References

[1] D. Kwon, J. H. Yoon, S.-Y. Shin, T.-H. Jang, H.-G. Kim, I. So, J.-H. Jeon and H. H. Park, "A comprehensive manually curated protein-protein interaction database for the Death Domain superfamily," *Nucleic Acids Research*, vol. 40, pp. D331-D336, 2012. Article (CrossRef Link)

[2] D. E. Gordon, G. M. Jang, M. Bouhaddou, J. W. Xu, K. Obernier, K. M. White, M. J. O'Meara, V. V. Rezelj, J. F. Z. Guo, D. L. Swaney et al, "A SARS-CoV-2 protein interaction map reveals targets for drug repurposing," *Nature*, vol. 583, pp. 459-468, 2020. Article (CrossRef Link)

[3] M. Altmann, S. Altmann, P. A. Rodriguez, B. Weller, L. E. Vergara, J. Palme, N. M. de la Rosa, M. Sauer, M. Wenig, J. A. Villaecija-Aguilar et al, "Extensive signal integration by the phytohormone protein network," *Nature*, vol. 583, pp. 271-276, 2020. Article (CrossRef Link)

[4] K. Yu, P.-Y. Lung, T. Zhao, P. Zhao, Y.-Y. Tseng and J. Zhang, "Automatic extraction of protein-protein interactions using grammatical relationship graph," *BMC Medical Informatics and Decision Making*, vol. 18, 2018. Article (CrossRef Link)

[5] S. Pyysalo, A. Airola, J. Heimonen, J. Björne, F. Ginter and T. Salakoski, "Comparative analysis of five protein-protein interaction corpora," *BMC Bioinformatics*, vol. 9, Article no. S6, 2008. Article (CrossRef Link)

[6] W. A. Baumgartner, Z. Lu, H. L. Johnson, J. G. Caporaso, J. Paquette, A. Lindemann, E. K. White, O. Medvedeva, K. B. Cohen and L. Hunter, "Concept recognition for extracting protein interaction relations from biomedical text," *Genome Biology*, vol. 9, Article no. S9, 2008. Article (CrossRef Link)
[7] G. Murugesan, S. Abdulkadhar and J. Natarajan, "Distributed smoothed tree kernel for protein-protein interaction extraction from the biomedical literature," *PLOS ONE*, vol. 12, pp. e0187379, 2017. Article (CrossRef Link)
[8] A. Airola, S. Pyysalo, J. Björne, T. Pahikkala, F. Ginter and T. Salakoski, "All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning," *BMC Bioinformatics*, vol. 9, 2008, Article no. S2. Article (CrossRef Link)
[9] Z. H. Zhao, Z. H. Yang, H. F. Lin, J. Wang and S. Gao, "A protein-protein interaction extraction approach based on deep neural network," *Int J Data Min Bioinform*, vol. 15, pp. 145-164, 2016. Article (CrossRef Link)
[10] C. Quan, L. Hua, X. Sun and W. Bai, "Multichannel Convolutional Neural Network for Biological Relation Extraction," *Biomed Res Int*, vol. 2016, no. 1850404, 2016. Article (CrossRef Link)
[11] L. Hua and C. Quan, "A Shortest Dependency Path Based Convolutional Neural Network for Protein-Protein Relation Extraction," *Biomed Res Int*, vol. 2016, no. 8479587, 2016. Article (CrossRef Link)
[12] Y. Peng and Z. lu, "Deep learning for extracting protein-protein interactions from biomedical literature," in *Proc. of The BioNLP 2017 workshop*, pp. 29-38, 2017. Article (CrossRef Link)
[13] S. P. Choi, "Extraction of protein-protein interactions (PPIs) from the literature by deep convolutional neural networks with various feature embeddings," *J Inf Sci*, vol. 44, pp. 60-73, 2018. Article (CrossRef Link)
[14] H. Zhang, R. C. Guan, F. F. Zhou, Y. C. Liang, Z. H. Zhan, L. Huang and X. Y. Feng, "Deep Residual Convolutional Neural Network for Protein-Protein interaction Extraction," *Ieee Access*, vol. 7, pp. 89354-89365, 2019. Article (CrossRef Link)
[15] Y. L. Hsieh, Y. C. Chang, N. W. Chang and W. L. Hsu, "Identifying Protein-protein Interactions in Biomedical Literature using Recurrent Neural Networks with Long Short-Term Memory," in *Proc. of The 8th IJCNLP*, pp. 240-245, 2017. Article (CrossRef Link)
[16] M. Ahmed, J. Islam, M. R. Samee, and R. E. Mercer, "Identifying Protein-Protein Interaction using Tree LSTM and Structured Attention," in *Proc. of IEEE 13th ICSC*, 2019. Article (CrossRef Link)
[17] S. Yadav, A. Ekbal, S. Saha, A. Kumar and P. Bhattacharyya, "Feature assisted stacked attentive shortest dependency path based Bi-LSTM model for protein-protein interaction," *Knowledge-Based Syst*, vol. 166, pp. 18-29, 2019. Article (CrossRef Link)
[18] J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proc. of NAACL*, Minneapolis, Minnesota, USA, pp. 4171-4186, 2019.
[19] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So and J. Kang, "BioBERT: a pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, pp. 1234-1240, 2020. Article (CrossRef Link)
[20] N. Warikoo, Y.-C. Chang and W.-L. Hsu, "LBERT: Lexically aware Transformer-based Bidirectional Encoder Representation model for learning universal bio-entity relations," *Bioinformatics*, vol. 37, pp. 404-412, 2021. Article (CrossRef Link)
[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, "Attention is all you need," in *Proc. of 31st ICNIPS*, Long Beach, California, USA, pp. 6000–6010, 2017.
[22] T. Yu, R. Jin, X. Han, J. Li and T. Yu, "Review of Pre-training Models for Natural Language Processing," *CEA*, vol. 56, no. 23, pp. 12-22, 2020. Article (CrossRef Link)
[23] T. Mikolov, I. Sutskever, K. Chen, G. Corrado and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," in *Proc. of NIPS*, 2013. Article (CrossRef Link)
[24] J. Pennington, R. Socher and C. Manning, "Glove: Global Vectors for Word Representation," in *Proc. of EMNLP*, 2014. Article (CrossRef Link)

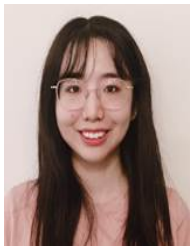
[25] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee and L. Zettlemoyer, "Deep Contextualized Word Representations," in *Proc. of NAACL*, New Orleans, Louisiana, USA, pp. 2227-2237, 2019.
[26] J. Howard and S. Ruder, "Universal Language Model Fine-tuning for Text Classification," in *Proc. of ACL*, Melbourne, Australia, pp. 328-339, 2018.
[27] R. Alec, N. Karthik, S. Tim and S. Ilya, "Improving Language Understanding by Generative Pre-Training," in *Proc. of NLPIR*, 2019. Article (CrossRef Link)
[28] H. Yang, J. Yuan, C. Li, G. Zhao, Z. Sun, Q. Yao, B. Bao, A. V. Vasilakos and J. Zhang, "BrainIoT: Brain-Like Productive Services Provisioning with Federated Learning in Industrial IoT," *IEEE Internet of Things Journal*, vol. 9, pp. 2014-2024, 2022. Article (CrossRef Link)
[29] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow and R. Fergus, "Intriguing properties of neural networks," in *Proc. of ICLR*, 2014. Article (CrossRef Link)
[30] J. Goodfellow, J. Shlens and C. Szegedy, "Explaining and Harnessing Adversarial Examples," in *Proc. of ICLR*, 2015. Article (CrossRef Link).
[31] T. Miyato, A. M. Dai and I. Goodfellow, "Adversarial Training Methods for Semi-Supervised Text Classification," in *Proc. of ICLR*, 2017. Article (CrossRef Link)
[32] A. Madry, A. Makelov, L. Schmidt, D. Tsipras and A. Vladu, "Towards Deep Learning Models Resistant to Adversarial Attacks," in *Proc. of ICLR*, 2018. Article (CrossRef Link)
[33] L. Ba, J. R. Kiros and G. E. Hinton, "Layer Normalization," *arxiv*, 2016. Article (CrossRef Link)
[34] M. Jian, K. M. Lam, J. Dong, et al, "Visual-Patch-Attention-Aware Saliency Detection," *IEEE Transactions on Cybernetics*, vol. 45(8), pp.1575-1586, 2015. Article (CrossRef Link)
[35] M. Jian, W. Zhang, H. Yu, et al, "Saliency Detection Based on Directional Patches Extraction and Principal Local Color Contrast," *Journal of Visual Communication and Image Representation*, vol.57, pp. 1-11, 2018. Article (CrossRef Link)
[36] M. Jian, J. Wang, H. Yu, et al, "Visual saliency detection by integrating spatial position prior of object with background cues," *Expert Systems with Applications*, vol. 168(11), pp. 114219, 2020. Article (CrossRef Link)
[37] M. Jian, Qi. Q ,J. Dong, et al, "Integrating QDWD with pattern distinctness and local contrast for underwater saliency detection," *Journal of Visual Communication and Image Representation*, vol. 53, pp. 31-41, 2018. Article (CrossRef Link)
[38] R. Bunescu, R. Ge, R. J. Kate, E. M. Marcotte, R. J. Mooney, A. K. Ramani and Y. W. Wong, "Comparative experiments on learning information extractors for proteins and their interactions," *Artificial Intelligence in Medicine*, vol. 33, pp. 139-155, 2005. Article (CrossRef Link)
[39] S. Pyysalo, F. Ginter, J. Heimonen, J. BjoRne, J. Boberg, J. Rvinen and T. Salakoski, "BioInfer: a corpus for information extraction in the biomedical domain," *BMC Bioinformatics*, vol. 8, 2007.
[40] Fundel, R. Kueffner and R. Zimmer, "RelEx - Relation extraction using dependency parse trees," *Bioinformatics*, vol. 23, pp. 365-371, 2007. Article (CrossRef Link)
[41] D. B. A'D, J. Da and D. Nb, "Mining Medline: Abstracts, Sentences, Or Phrases?," in *Proc. of Pacific Symposium on Biocomputing Pacific Symposium on Biocomputing*, vol. 7, pp. 326-337, 2002.
[42] C. Nédellec, ''Learning language in logic-genic interaction extraction challenge,'' in *Proc. of Learn. Lang. Logic Workshop*, pp. 1–7, 2005.
[43] Y.-C. Chang, C.-H. Chu, Y.-C. Su, C. C. Chen and W.-L. Hsu, "PIPE: a protein–protein interaction passage extraction module for BioCreative challenge," *Database*, vol. 2016, no. baw101, 2016. Article (CrossRef Link)

ZHAN TAN received the M.S. degree in computer science from Shanghai Normal University and his Bachelor's degree from Southwest Jiaotong University. He is currently pursuing the Ph.D. degree with the College of Information and Electrical Engineering, China Agricultural University, Beijing, China. His research interests include natural language processing, text mining, and deep learning.

ZHAO BAI received a bachelor's degree in engineering from Anhui University of Science and Technology. He is currently pursuing the M.S. degree with the College of Information and Electrical Engineering, China Agricultural University, Beijing, China. His research interests is computer vision.

LEI DIAO received a bachelor's degree in engineering from Anhui University of Science and Technology. He is currently pursuing the M.S. degree with the College of Information and Electrical Engineering, China Agricultural University, Beijing, China. His research interests include natural language processing, text mining, and deep learning.

XUCHAO GUO received his Bachelor's degree in Computer Science and his Master's degree in 2018 from Shandong Agricultural University. He is currently Ph.D. student in the College of Information and Electrical Engineering, China Agricultural University. He is mainly engaged in natural language processing and knowledge graph in agricultural fields. His research interests include: deep learning, complex network analysis, data mining, machine learning, and image processing.



SHUHAN LU received her Bachelor's degree in Computer and Information Science in 2020 at the Ohio State University. From 2020, she is currently studying Master's in Health Informatics in University of Michigan, Ann Arbor. She has good experience in database creating, database managing, and data analysis. Her research interests include: data mining, database manage, machine learning and image processing.

LIN LI is currently a Professor and a Doctoral Supervisor with the College of Information and Electrical Engineering (CIEE), China Agricultural University. Her main research interests include knowledge engineering and machine learning.