Protein Transformer CPI: A Submodel Enhancing Protein Representation Learning in Compound Protein Interaction Prediction

Jingtao Wang 1,2,*, Xi Li 3

1 Computational Biology Department, School of Computer Science, Carnegie Mellon University
2 SenseTime
3 College of Computer Science, Zhejiang University

Abstract
Motivation: Machine learning methods for predicting compounds protein interactions (CPIs) are crucial for the in-silico screening step of drug discovery. In recent years, many end-to-end representation learning methods using deep neural networks have achieved significantly better performance than traditional machine learning algorithms. Much effort has been paid on improving the capability of the model by taking advantage of neural attention mechanism, either in learning compound representation, or learning the interaction between protein representation and compound representation. However, seldom has been done to improve the protein representation learning, while current approaches have manifest flaw of lacking the ability of learning amino acids’ long-distance interactions, which are essential for determine the proteins’ properties due to protein folding.

Results: The authors propose a novel approach for incorporating self-attention in the protein representation learning module of CPI models, providing the module with the capability of capturing long-distance interaction information within proteins. And such approach can be universally applied to nearly any deep learning models or submodels for protein representation learning. By applying such approach on an existing CPI model, the modified version, with our proposed Protein Transformer as its protein learning module, has a significant improvement in the prediction performance. Practical tips for training Protein Transformer are also provided.

Availability: https://github.com/JingtaoWang22/CPI_prediction
Contact: jingtao2@cs.cmu.edu

1 Introduction

1.1 Background and Problem Setting
In-silico screening, which generates drug candidates, is the very first step of drug discovery. Machine learning-based methods predicting the compound-protein interactions (CPIs) have been playing an important role in this step. In the past decade, end-to-end representation learning using deep neural networks, which does not use any fixed feature, for discrete symbolic data (e.g., words in natural language processing) has demonstrated excellent performance on various difficult problems, and has been applied on the CPI problem1-8.

For the CPI problem, data are provided as discrete symbolic data, i.e., compounds are represented as graphs where the vertices are atoms, the edges are chemical bonds, and proteins are sequences in which the characters are amino acids.

1.2 General Framework
The deep learning models for CPIs are usually composed of 3 submodels: Compound-module (C-module), the part of neural networks learning the representation of compounds; Protein-module (P-module), the networks learning the representation of proteins; Interaction-module (I-module), the networks take the output of the previous 2 modules and output the final prediction.
The deep learning models for the C-module and P-module needs to be compatible with the data structure of the compound (graph) and protein (sequence) respectively. Therefore, a common choice is using a Graph Neural Networks (GNN) for the C-module, and a Convolutional Neural Networks (CNN) or recurrent neural network (RNN) for the P-module. For example, the first end-to-end representation learning model proposed by Tsubaki et al. adopts GNN for the C-module and CNN for the P-module. The outputs of C-module and P-module are the vector representations of compounds and proteins respectively, and they will be concatenated and sent to the I-module for predicting the interaction.

1.3 Applying Neural Attention Mechanism
In recent years, many variations of attention mechanisms have been applied to capture the interactions of vector representations in different fields including computer vision, nature language processing (NLP), biological data analysis, etc. For example, Transformer is a nature language translation model, which achieved state-of-the-art performance at the time, by relying solely on attention mechanism (no CNN or RNN). In particular, this model adopts self-attention mechanism as a substitution of RNN due to its capability of capturing the interaction between words representations regardless of long distance.

The I-module in the CPI framework also often adopts neural attention mechanism to predict the interaction of compounds and proteins. A good usage of the attention mechanism is often the key to predicting the interaction and finding the binding sites between compounds and proteins. Some novel models take a step further and incorporate self-attention in their I-module based on the intuition that taking mutual interaction between every dimension of compound and protein representation into consideration helps the model makes a more comprehensive decision.

Molecule Transformer DTI, instead, utilizes self-attention in their C-module, i.e. learning compound representation, to better capture the mutual interaction between atoms in the compounds. They argue that self-attention can better relate long distance atoms in chemical compounds better than other networks.

1.4 Improving P-module Using Self-Attention (The Authors’ Work)
In this work, the authors propose a novel approach to ameliorating the P-module in the CPI framework by taking advantage of the self-attention mechanism, and demonstrate its effectiveness using a series of control experiments. We successfully improved the P-module of Tsubaki et al’s model by capturing the information of long-distance interaction within protein, including protein folding, which is essential for deciding protein’s chemical and biological functions. Such interactions are impossible to be caught by CNNs or RNNs. However, we achieved this by implementing self-attention encoding layer. And we show that the modified version of the model, with Protein Transformer as P-module, has a significant improvement in terms of AUC, precision and recall on the same datasets. Additionally, the new model is also very universal: it uses the same set of hyperparameters (learning rate, radius, n-gram, number of different layers, etc.) for all datasets, indicating its strong ability of modeling the true mechanisms underlying the compound protein interactions.
Apart from the new approach and the model, we are also giving some practical issues in training the Protein Transformer regarding the learning rate, model dimension, number of self-attention encoding layers, and other hyperparameters. Most importantly, we found that although self-attention is effective in modeling protein properties like folding, CNN layers are still important because they are better at extracting the features directly relevant to CPI and capture the local interactions that are ‘diluted’ by self-attention encoding layers.

The authors also argue that our method can be universally applied to near any deep learning model involving learning the representation of proteins. Self-attention encoding layers can be added to the bottom of the networks for learning protein representation, so that information about the long-distance interactions in protein can be incorporated in the learning. And it is almost always worth doing so because folding is a crucial component underlying the protein’s many properties.

2 Motivation

2.1 Long-distance Dependencies

However, while most of the efforts have been paid on improving the C-module (compound learning) and the I-module (interaction prediction) of the CPI model as discussed in section 1.3, the P-module (protein learning) lacks the attention it deserves. Current models adopt RNNs and CNNs for the P-module, which leads to losing the ability of capturing the long-distance interaction within proteins.

RNNs and many of their variations (e.g. LSTM) have long been criticized for having a short-term memory and not being able to learn the long-distance dependencies in sequences. CNNs’ nature also determines its lack of such ability. For example, 3 layers of CNNs with filter size 11 in Tsubaki et al.’s model can merely catch the association within 31 amino acids’ long, while the total length of a protein can be thousands amino acids. However, such long-distance associations in proteins are crucial for deciding the proteins’ properties. In fact, the long-distance associations in proteins can potentially be more informative than those in the C-module or I-module. This is not only because proteins are sequences much longer than compounds or representations in the I-module, but also due to more long-distance interactions in proteins, and the very important roles they play. For instance, although 2 amino acids can be seemingly far away from each other on the sequence, they could in fact have a very short spatial distance due to protein folding, and their interaction can further have impact on the folding. ‘Structure determines function’ is a key idea in biology, indicating such mutual interaction between folding and long-distance amino acids associations can be very informative in terms of predicting proteins’ functions.

2.2 The Self-Attention Mechanism

Vaswani et al. adopted self-attention as a substitution of RNNs because it is more capable of learning the long-distance dependencies between words. In fact, self-attention learns dependencies regardless of distance, which also has its downside of losing positional information. And positional encoding is incorporated for dealing with this issue. In our proposed Protein Transformer, self-attention is the core component for addressing this long-distance dependency
An attention function is a mapping from a Key-Value (K-V) pair and a Query (Q) to an output, where the Query, Key, Value, and the output are all vector representations. In our case, Q, K, V are the linear projection of the same input protein sequence representations, and the output is the new protein representation incorporating the mutual association between amino acids. The whole process includes three steps: Acquiring the linear projections Query Key Value; computing the Weight by putting the Query and the Key into a compatibility function; and getting the output by computing the weighted sum of the Value using the computed Weight as the weight.

The compatibility function has many kinds of variations making there to be many versions of attentions. In this work, due to the length of the protein data (thousands of amino acids), the authors adopt the version with least time and space efficiency: “Scaled Dot-Product Attention”. The compatibility function of it calculates the dot product of the Query and the Key, divides it by $\sqrt{d_k}$ where $d_k$ is the dimension of Key, and finally apply softmax on it to get the Weight.

$$\text{Weight} = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)$$  \hspace{1cm} (1)

Here, Weight is a square matrix with the number of rows/columns equal to the length of the protein, i.e. the number of amino acids. The value in the i-th row j-th column of the Weight represents the interaction intensiveness between the i-th and j-th amino acids.

After calculating the weight, each row of the output, which is an amino acid vector, can be calculated as the weighted sum of all amino acids. This is accomplished by a single matrix multiplication:

$$\text{Output} = \text{Weight} \times V = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$ \hspace{1cm} (2)

The intuition behind this is that amino acids should be allowed to interact based on the strength of the mutual interactions.

3 Model

3.1 The Overall Architecture
On the left of the picture shows the C-module taking molecule embedding as input and it outputs compound representation vector. This compound representation will be used in the P-module for guiding protein representation learning, and in I-module for the final interaction prediction. On the right is the P-module for learning the protein representation. It takes protein embedding and compound representation, and outputs protein representation. Finally, on top of the C-module and P-module, I-module takes the output of them, and produces the final interaction prediction.

The protein embedding will be attached with positional encoding first, and then fed into N self-attention encoders for learning the long-distance interactions (mainly folding information).

Protein sequence will also be preprocessed into $d_m$ dimensional space and fed into a “protein transformer”. The output, the protein vector, will concatenate with the learned compound vector and go through a linear layer, and finally use softmax to predict the final interaction.

### 3.2 C-module for Compound Representation Learning

The C-module is identical to Tsubaki et al.’s model because this is a part of the control in our experiments. We want to show that the improvement of the model comes from incorporating the self-attention mechanism rather than any other modification of the model.

In the C-module, compound molecules will be embedded into $d_m$ dimensional space and fed into a 3-layer GNN. The output of the GNN is the compound representation (a $d_m$ dimensional vector).
3.21 Preprocessing of the compound molecule inputs

Before being fed into GNN, compound data, which are graphs consisting of atoms as vertex and chemical bonds as edges, are preprocessed and embedded. During preprocessing, within each compound molecule $G_C = (V, E)$, nodes are updated as fingerprints according to their r-radius subgraph. Subsequently, fingerprints are embedded into $d_m$ dimensional space and fed into the GNN together with the adjacency matrices to generate compound representations.

Here is the detail of the fingerprint generation:
For any input compound $G_C = (V, E)$, given hyperparameter radius $r$, we update each vertex $v \in V$ for $r$ iterations. In each iteration, vertices are updated by its neighbors:

$$v \rightarrow (v, (v_1, v_2, ..., v_n)) \quad (3)$$

Where $v_1, v_2, ..., v_n \in \text{neighbors}(v)$

Thus, after $r$ iterations, each node becomes a representation of a r-radius subgraph with itself as the center.

3.22 GNN for Compound representation learning

Given hyperparameter $t$, layers of GNN, input $G_C = (V, E)$ is updated for $t$ iterations, while in each iteration:

$$H_t = \text{ReLU}(W_t X_C(t) + B_t) \quad (4)$$
$$X_C(t+1) = X_C(t) + AH_t \quad (5)$$

Where in (4):
$H_t \in \mathbb{R}^{|V| \times d}$ is the hidden neighborhood vector of the $t$-th layer of the GNN. $W_t \in \mathbb{R}^{|V| \times |V|}$ and $B_t \in \mathbb{R}^{|V| \times d}$ are weight and bias of the linear transformation of the input vertex feature matrix

$$X_C(t) = \begin{bmatrix} v_1^{(t)} \\ v_2^{(t)} \\ \vdots \\ v_{|V|}^{(t)} \end{bmatrix} \in \mathbb{R}^{|V| \times d}$$

ReLU, originally proposed in [5], is a non-linear activation function: $\text{ReLU}(x) = \max(0, x)$.

In (5):
$A \in \mathbb{R}^{|V| \times |V|}$ is the adjacency matrix of $G_C$.

Thus, after $t$ iterations, nodes are updated according to the substructures of $G_C$ and $X_C$ becomes the learned representation of the compound. It is utilized by the attention mechanism of the Protein Transformer to learn the protein representation $X_P$ as well as finding the interaction site between protein and compound. Finally, after the protein representation is derived, $X_C$ is also concatenated with $X_P$ to predict the interaction.

3.3 Protein Transformer as the P-module for Protein Representation Learning

In this section we propose our approach to improving the original P-module by taking advantage of self-attention, and present our new submodel Protein Transformer, which is a universal model that can be used in other deep learning model or submodel for learning protein representation.

Protein Transformer uses self-attention encoding layers to extract proteins information regarding folding and other long-distance interactions, and then applies attention CNN to learn binding site
representation, which is also the final output of the protein vector.

### 3.31 Preprocessing of the protein sequence inputs
Initially, proteins are provided as amino acid sequences and need to be preprocessed:

\[ PS = A_2A_3 \ldots A_{|s|} \]

where \( A_i \) is the i-th amino acid.

They are first converted into n-gram words:

\[ S = w_1w_2w_3 \ldots w_{|s|} \]

where \( w_i = A_{i-1}A_iA_{i+1} \). We set \( A_0 = \text{'}-\text{'} \) & \( A_{|s|+1} = \text{'}=\text{'} \) to be the starting & ending symbol respectively. Here we set \( n=3 \) which is also identical to Tsubaki et al's model.

Then \( S \) is embedded into \( d_m \) dimensional space as the protein input:

\[
X_P = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{|s|} \end{bmatrix}
\]

Where \( x_i \in \mathbb{R}^{d_m} \) is the \( d_m \) dimensional embedding of \( w_i \).

After these preprocessing steps, protein input \( X_P \in \mathbb{R}^{|s| \times d} \) is fed into the “Protein Transformer” to learn the protein representation vector.

### 3.32 Protein Transformer for Protein Representation Learning
The protein sequence preprocessed into \( X_P \in \mathbb{R}^{|s| \times d} \) is fed into the Protein Transformer. Figure 2 shows the general structure of the P-module, which is the Protein Transformer. And figure 3 and 4 shows the detailed structure of the encoder and decoder component respectively.

**Positional Encoding**
Before feeding into self-attention layer, the representation must be attached with positional encoding. The self-attention mechanism in our proposed Protein Transformer acquires the ability of learning long distance dependencies at the expense of totally ignoring the positional information of tokens. Therefore, same as the Transformer, positional information of each token is attached to \( X_P \):

\[
\text{PE}(\text{pos}, 2i) = \sin \left( \frac{\text{pos} \cdot 10000^{2i}}{10000^{2d_m}} \right) \quad (6)
\]
\[ PE(pos, 2i + 1) = \cos \left( \frac{pos}{10000^{2i}} \right) \]  
\[ X_p = X_p + PE \]  
Where, \( PE(pos, i) \) is the position encoding of the token at position \( pos \) and dimension \( i \), and \( d_m \) is the dimension of the model inputs.

The Self-Attention Encoding Layer
The self-attention encoding layers are the key component for modeling long-distance interactions, including folding, within proteins. The authors demonstrate that incorporating self-attention encoder can help the model learn the protein representation better, which leads to a significant improvement in the final prediction performance.

Preprocessed protein vector \( X_p \) with positional encoding is fed into \( N \) encoders, where \( N \), the number of encoders, is a hyperparameter. In each encoder, \( X_p \) goes through a Multi-Head Self-Attention layer and a Feed Forward layer with Residual Connection.

\[ X_p \in \mathbb{R}^{s \times d_m} \] first goes into the Multi-Head Attention Layer, where it is divided into \( h \) channels: \( X_{pl} \in \mathbb{R}^{s \times d_k} \), where \( i = 1, 2, ..., h \), and \( h \cdot d_k = d \). For each channel, \( X_{pl} \), a lower dimensional representation of protein, goes through a self-attention layer, in which \( X_{pl} \) is the Key, Query, Value:

\[ AttentionScore = Weight = \text{Softmax} \left( \frac{Query \cdot Key^T}{\sqrt{d_k}} \right) \]  
\[ X_{pl} = AttentionScore \times X_{pl} \]  
\[ X_p = [X_{p1}, ..., X_{ph}] \]  
As illustrated in section 2.2, self-attention first calculates a weight representing the strength of mutual interactions (9), and then uses this weight to let amino acid representations in \( X_{pl} \) interact with each other (10). Thus, the output acquires the information of interaction.

Multi-Head Attention project \( X_{pl} \) into different subspaces, making them to learn different aspects of the protein’s properties. After the attention layer \( X_{pl} \)’s are concatenated and outputted.

The Multi-Head Attention is used with Residual Connection:

\[ X_p = \text{Normalization}(X_p + \text{MultiHeadAttention}(X_p)) \]  
This is followed by a Feed Forward layer with Residual Connection:

\[ X_p = \text{Normalization}(X_p + W_{FF}X_p + B_{FF}) \]  
In summary, the self-attention encoding layer, i.e. the self-attention encoder, takes a protein representation vector, applies multi-head attention to learn multiple aspects of protein’s self-interaction properties, and wraps it with a residual connection so that the network can decide whether to skip the current layer.

CNN
Although self-attention is an effective mechanism for capturing the mutual association between
amino acids, such information is not sufficient for predicting CPI because the properties essential for binding with compounds are not necessarily learned. Our experiments also show that a P-module with merely attention mechanisms cannot yield satisfactory results. We regard this as a very tricky part in designing Protein Transformer. CNN is still crucial for learning the binding site properties of protein. Therefore, the self-attention mechanism should act more like a supplementary for the P-module. It serves as a sub network for capturing a certain category of information, which is long-distance association. And it should be learned jointly with other network suitable for capturing the property relevant to binding with compounds.

Intuitively, CNN’s filter is very suitable for modeling the binding of CPIs. It scans along the protein sequence searching for the subsequence matching the certain patterns indicating a binding site. Thus, we believe attention CNN replenished by the self-attention encoder should turn out to be an effective model capable of learning all aspects of protein’s properties.

Thus, in Protein Transformer, after $N$ layers of Self-Attention encoders, $X_P$ is fed into an attention CNN sub network, which is 3 layers of CNN.

**Target Attention Decoder**

After CNN, the representation is fed into the decoder for learning target relevant information, where the target of course is the compound representation. Decoder shares a very similar structure with the decoder in Transformer. Compound target will be fed into a target-attention layer to interaction with the protein representation. The target-attention is very similar to self-attention, while the *Query* is the compound representation $[X_c \ldots X_c] \in \mathbb{R}^{|s| \times d_m}$, which is a matrix consists of $|s|$ atom representations stacked vertically. The *Key, Value* are still $X_P$, the protein representation. Thus “words” in $X_P$ are weighted according to their interaction intensity with the compound, instead of the interaction intensity within the protein as in the self-attention layer.

After that, the output is fed into an encoder-layer-like structure and sent through a final target attention layer where the length of the representation will also be squeezed to one. The representation will be sent to the I-module for interaction prediction after all its process in the P-module.

**3.4 I-module for Interaction Prediction**

The last step of the model is predicting the interaction. $X_P \in \mathbb{R}^{|p| \times d_m}$ and $X_C \in \mathbb{R}^{|v| \times d_m}$ are both summed up along the first dimension:

$$ For \ X_C = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_{|v|} \end{bmatrix}, X_C = [v_1 + v_2 + \ldots + v_{|v|}], X_C^T \in \mathbb{R}^{d_m} \quad (14) $$

$$ For \ X_P = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_{|s|} \end{bmatrix}, X_P = [X_1 + X_2 + \ldots + X_{|s|}], X_P^T \in \mathbb{R}^{d_m} \quad (15) $$
Then use Softmax to predict the probability of interaction:

\[ Z = W_{\text{Output}} \begin{bmatrix} X_C \\ X_P \end{bmatrix} + B_{\text{Output}} = [y_0, y_1] \]  
(16)

\[ P_t = \frac{\exp(y_t)}{\sum_{i=0,1} \exp(y_i)} \]  
(17)

Where \( t = 0/1 \) stands for interacting or not.

In training, the cross-entropy loss is used:

\[ L = -\sum_{i=1}^n T_i \log P_{ti} \]  
(18)

Where \( n \) is the total number of compound-protein pairs, \( T_i \) is the ground-truth of the \( i \)-th pair, \( P_{ti} \) is the probability of the true label.

4 Results

4.1 Experimentations

The authors run the model on 2 datasets (human & C.elegans) created by Liu et al., which is also used by Tsubaki et al. We used all same hyperparameters for the datasets demonstrating that our model is universal for all kinds of CPI data. We used number of encoding layer = 3, number of decoding layer = 1, learning rate = 2e-4, warmup step for the attention mechanism = 50 for all datasets. The performance of our model is recorded in the last column (GNN-PT).

**Human positive/negative ratio 1:1**

| Measure   | k-NN  | RF   | L2   | SVM  | Tsubaki et al.'s | GNN-PT |
|-----------|-------|------|------|------|------------------|--------|
| AUC       | 0.860 | 0.940 | 0.911 | 0.910 | 0.970            | 0.978  |
| Precision | 0.798 | 0.861 | 0.891 | 0.966 | 0.923            | 0.928  |
| Recall    | 0.927 | 0.897 | 0.913 | 0.950 | 0.918            | 0.952  |

**Human positive/negative ratio 1:3**

| Measure   | k-NN  | RF   | L2   | SVM  | Tsubaki et al.'s | GNN-PT |
|-----------|-------|------|------|------|------------------|--------|
| AUC       | 0.904 | 0.954 | 0.920 | 0.942 | 0.950            | 0.980  |
| Precision | 0.716 | 0.847 | 0.837 | 0.969 | 0.949            | 0.917  |
| Recall    | 0.882 | 0.824 | 0.773 | 0.883 | 0.913            | 0.925  |

**C.elegans positive/negative ratio 1:1**

| Measure   | k-NN  | RF   | L2   | SVM  | Tsubaki et al.'s | GNN-PT |
|-----------|-------|------|------|------|------------------|--------|
| AUC       | 0.858 | 0.902 | 0.892 | 0.894 | 0.978            | 0.983  |
| Precision | 0.801 | 0.821 | 0.890 | 0.785 | 0.938            | 0.943  |
| Recall    | 0.827 | 0.844 | 0.877 | 0.818 | 0.929            | 0.936  |

**C.elegans positive/negative ratio 1:3**

| Measure   | k-NN  | RF   | L2   | SVM  | Tsubaki et al.'s | GNN-PT |
|-----------|-------|------|------|------|------------------|--------|
| AUC       | 0.892 | 0.926 | 0.896 | 0.901 | 0.971            | 0.983  |
| Precision | 0.787 | 0.836 | 0.875 | 0.837 | 0.916            | 0.942  |
| Recall    | 0.743 | 0.705 | 0.681 | 0.576 | 0.921            | 0.929  |
The results of k-NN, Random Forest, L2 Logistic, Support Vector Machine, are reported in Liu et al.’s work\textsuperscript{14}. GNN-Protein Transformer is the model designed by the authors. As shown above, GNN-PT outperforms the existing methods on almost all conditions in terms of metrics and datasets. Notably, our model has a significantly higher AUC in all conditions, demonstrating this model learn the knowledge very well and the prediction results are always aligning with the truth.

Notice that the main contribution of our work is proposing this approach of incorporating self-attention to the P-module to help the model learn better, and this approach can almost be applied to any model for protein sequences. Based on this idea, and also because the authors are not available to much computational resources, we did not try to fine-tune the model to get the optimal results. Our argument has been justified since the current one has an significant improvement. That said, we discovered phenomena like stacking more self-attention encoder tend to further improve the performance, etc. Those are worth trying so that a better model can be discovered.

References:
[1] Tsubaki, M., Tomii, K., and Sese, J. (2019). Compound–protein interaction prediction with end-to-end learning of neural networks for graphs and sequences. Bioinformatics 35, 309–318.
[2] Kyle Yingkai Gao, Achille Fokoue, Heng Luo, Arun Iyengar, Sanjoy Dey, and Ping Zhang. Interpretable drug target prediction using deep neural representation. In IJCAI, pages 3371–3377, 2018.
[3] Wan, F., Zhu, Y., Hu, H., Dai, A., Cai, X., Chen, L., Gong, H., Xia, T., Yang, D., Wang, M.W., and Zeng, J. (2019). DeepCPI: A deep learning-based framework for large-scale in silico drug screening. Genomics Proteomics Bioinformatics 17,478-495
[4] M Karimi, D Wu, Z Wang, and Y Shen. DeepAffinity: Interpretable deep learning of compound-protein affinity through unified recurrent and convolutional neural networks. Bioinformatics (Oxford, England), 2019.
[5] Lifan Chen, Xiaoqin Tan, Dingyan Wang, Feisheng Zhong, Xiaohong Liu, Tianbiao Yang, Xiaomin Luo, Kaixian Chen, Hualiang Jiang, Mingyue Zheng, TransformerCPI: improving compound–protein interaction prediction by sequence-based deep learning with self-attention mechanism and label reversal experiments, Bioinformatics, , btaa524
[6] Brighter Agyemang, Wei-Ping Wu, Michael Yelpengne Kpiebaareh, Zhihua Lei, Ebenezer Nanor, Lei Chen. Multi-View Self-Attention for Interpretable Drug-Target Interaction Prediction. arXiv:2005.00397v2
[7] Bonggun Shin, Sungsoo Park, Keunsoo Kang and Joyce C. Ho. Self-Attention Based Molecule Representation for Predicting Drug-Target Interaction. Proceedings of Machine Learning Research 106:1–18, 2019
[8] Shuya Li, Fangping Wan, Hantao Shu, Tao Jiang, Dan Zhao, Jianyang Zeng. MONN: a Multi-Objective Neural Network for Predicting Pairwise Non-Covalent Interactions and Binding Affinities between Compounds and Proteins. RECOMB 2020
[9] Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2009). The graph neural network model. IEEE Transactions on Neural Networks, 20(1), 61–80.
[10] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, q., and Polosukhin, I. (2017). Attention is all you need. In Advances in Neural Information Processing
[11] Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. ICLR.

[12] Dong, Q.-W., Wang, X.-L., and Lin, L. (2006). Application of latent semantic analysis to protein remote homology detection. Bioinformatics, 22(3), 285–290.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 770–778, 2016.

[14] Liu, H., Sun, J., Guan, J., Zheng, J., and Zhou, S. (2015). Improving compound–protein interaction prediction by building up highly credible negative samples. Bioinformatics, 31(12), i221–i229.