**A machine learning model employed bilinear attention mechanism for predicting drug-target affinity based on substructures**

## **Abstract**

In drug discovery, drug-target affinity is an important criterion for measuring whether the drug molecule is effective. In recent years, predicting affinity models by machine learning methods has become mainstream, but most of which do not pay attention to the substructures of drugs and targets. So this paper presents an innovative drug-target affinity prediction model, named substructure-based drug-target affinity bilinear attention Prediction model (SAB).

This model combines the advantages of graph convolutional networks (GCNs) and convolution neural networks (CNNs), and employs a bilinear attention mechanism to enhance the model’s ability to represent drug-target interactions, aimed at improving the accuracy of predictions deeply analyzing the substructural characteristics of drugs and targets.

In experimental design, the SAB model has extensively evaluated the affinity on multiple public datasets. These datasets cover both regression and classification tasks. Through this approach, the SAB model provides a new perspective on drug-target affinity prediction, emphasizing the importance of substructural features in the prediction process and showing the power of the bilinear mechanism in capturing drug-target interactions. This method not only improve the accuracy of predictions but also provide new tools for drug discovery and target identification in the future.

## **Introduction**

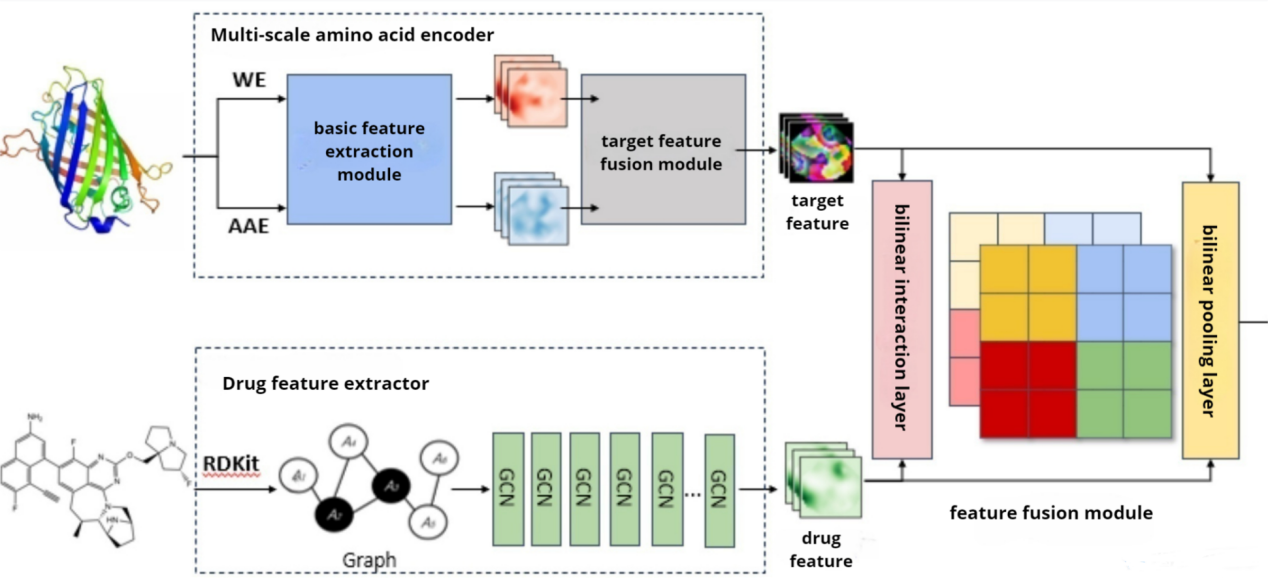
Among the existing models for predicting the affinity between drugs and target proteins, the two most famous ones are Sequence-based drug-target Affinity Prediction model (SAP) and Graph-based drug-target Affinity Prediction model (GAP). SAP primarily uses the text embeddings of drugs and targets as input to predict drug-target affinity, while GAP uses the molecular graph of the drug.

SAP mainly deals with the text sequence data of drugs and target proteins, which can be achieved by one-hot encoding or word embedding technology, converting original text sequences into a format that can be processed by machine learning or deep learning models.Then, sequence processing models such as Recurrent Neural Networks (RNN), Long Short-Term Memory networks (LSTM), or Convolution Neural Networks (CNN) are employed to extract features and predict affinity. This model is relatively simple but may not fully utilize the three-dimensional structural information of the drug molecules.

Compared with the SAP model, the GAP model directly uses graph data to represent drug molecules, where nodes represent atoms and edges represent chemical bonds. The algorithms it uses typically includes Graph Neural Networks (GNN), Graph Convolutional Networks (GCN), or Graph Attention Networks (GAT). The advantage of this model is that it can directly utilize the graph structure information of drug molecules, capturing richer chemical features, but it may need to handle more complex graph data, and require more complex model architecture.

However, whether it is SAP or GAP model, most of them uses the overall feature of drugs and targets. Therefore, the substructures of drug-target complexes are not given enough attention. These substructures play a key role in drug-target complex, affecting the binding affinity between drugs and targets. For example, in drug-target complexes, it is not the entire structure of drugs and targets that interacts with each other, but specific domains or motifs bind to particular groups of the ligand, while traditional prediction models often focus on overall features, ignoring these key substructures, leading to insufficient prediction accuracy and poor model explainability. So in order to predict affinity more accurately, it is necessary to consider and fully utilize the substructure information.

Given that point, we propose the SAB model. It not only considers the features of substructures but also integrates machine learning methods such as graph convolutional structure and bilinear attention mechanism. The overall architecture of the SAB model is shown in Figure 1. This model consists of three parts: a drug feature extractor, a multi-scale amino acid encoder, and a feature interaction module based on bilinear attention mechanism.



**Figure 1.** The proposed structure

The drug feature extractor consists of three multi-scale blocks and three transition layers. Multi-scale blocks can process features of different scales simultaneously in the network, while transition layers are used to smoothly transform features between different stages of network, reducing the number of parameters and computational complexity. We add a transition layer between two consecutive multi-scale blocks to integrate the features from the preceding multi-scale block. Each multi-scale block contains N convolutional layers[1]. There are various variants of convolutional layers, and here we use the GCN type: it updates the features of each node by a shared weight matrix and aggregates the information of neighboring nodes through the adjacency matrix of features.

The data input into the GCN layer comes from the Simplified Molecular Input Line Entry System (SMILES) strings of the drugs. We use the RDKit software[2]to convert it into molecular graph, and then input it into the drug feature extractor to extract the drug features.

The multi-scale amino acid encoder consists of a basic feature extraction module and a target feature fusion module. The basic feature extraction module includes a global feature extractor and a local feature extractor. They are both composed of CNNs, which convert the target amino acid sequence into two embedding vectors by amino acid embedding (AAE) coding and word embedding (WE) coding[3], and then extract target features. The feature fusion module, includes a fully connected layer and a Rectified Linear Unit (ReLU) layer, where the fully connected layer is used for feature integration, and the ReLU layer serves as an activation function, providing the network with nonlinear capability. Through the feature fusion module integrating these two features from global feature extractor and local feature extractor, the final target feature is output.

The feature fusion module based on the bilinear attention mechanism is composed of the bilinear interaction layer and the bilinear pooling layer. It firstly captures the attention weights of drug-target pairs and extracts the interactions between drug-target features[4] through the bilinear function, then reduces the spatial dimensions of the feature map through pooling operations, retaining important feature information simultaneously. Input the drug/targets features extracted previously into this module, obtaining the interaction information between these two, then the drug-target affinity prediction result can be output.

1. **Methods and Materials**

**Dataset preparation.** We evaluate model performance on Davis[5], KIBA[6] as regression tasks, and Human[7], BingdingDB[8], BioSNAP[9] as classification tasks. The statistics of the datasets are shown in Table 1. For each dataset, we randomly split it into training, testing, and validation sets at a ratio of 8:1:1.

**Table 1. Statistics of classification and regression task datasets**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task type | dataset | drug | Target protein | interaction pair |
| regression task | Davis | 68 | 442 | 30,056 |
|  | KIBA | 2,111 | 229 | 118,254 |
| classification task | Human | 2,726 | 2,001 | 6,728 |
|  | BingdingDB | 14,643 | 2,623 | 49,199 |
|  | BioSNAP | 4,510 | 2,181 | 27,464 |

*The dataset used for the regression task.*In the regression task, the datasets used are Davis and KIBA, as shown in Table 2. These two datasets are commonly used for evaluating binding affinity predictions in Drug-Target Interaction (DTI) studies, where the drugs are mostly small molecules with less than 100 atoms, and the target amino acid sequences are mainly less than 1500 characters, which follow a relatively normal distribution[10].

**Table 2 Specific information of the Davis and KIBA datasets**

|  |  |  |
| --- | --- | --- |
| Information | Davis | KIBA |
| Number of target proteins | 442 | 229 |
| Number of drugs | 68 | 2111 |
| Total sample | 30056 | 118254 |
| Active sample | 2457 | 22729 |
| Inactive sample | 27599 | 95525 |
| Number of training set samples | 24045 | 94603 |
| Number of test set samples | 3006 | 11831 |
| Number of validation set samples | 3005 | 11830 |

Davis comprises 72 drugs and 442 targets’ binding affinities, measured by dissociation constant (Kd), ranging from 5.0 to 10.8. The values used in actual modeling are pKd[11] values transformed into log space, similar to the data processing methods by He et al.[12] as shown in formula (1). The label distribution in the dataset is imbalanced, with the pKd values in 5~6 taking up more than half of the data in this dataset.

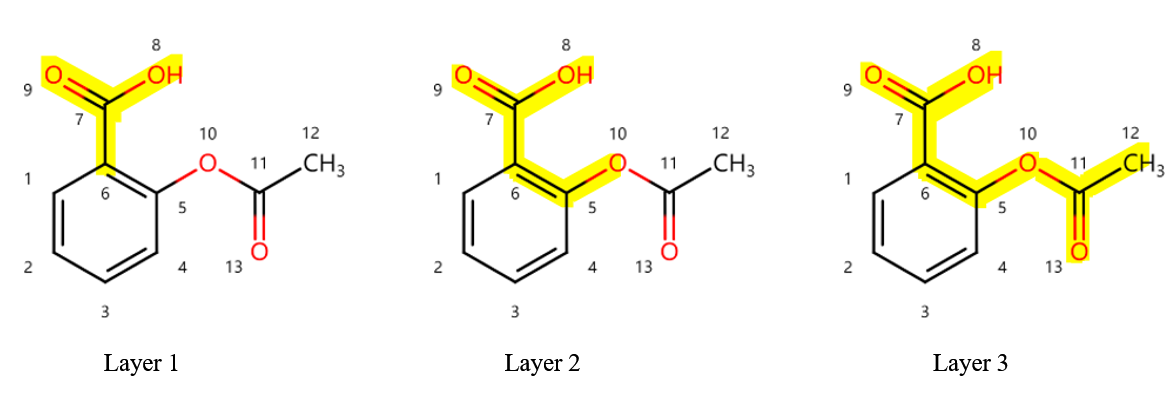
Kiba comprises 2116 drugs and 229 targets’ binding affinities, measured by Korea Institute of Bioinformatics Alliance (KIBA) scores, which optimizes the consistency of inhibition constant (Ki), Kd, and inhibitory concentration 50 (IC50) through the statistics information contained in them, ranging from 0.0 to 17.2[12].

*The dataset used for the classification task.*In the classification task, the datasets used are Human, BingdingDB, and BioSNAP, where 1 indicates " interaction" and 0 indicates "no interaction". The Human dataset contains experimentally verified positive samples and highly reliable negative samples obtained through computer simulation screening methods, totally 726 drugs and 2001 target interactions, constructed by Liu et al.[7]. The BingdingDB database is an open online database mainly including interactions and binding affinity data between small molecule drugs and targets. Based on the previous BingdingDB 2015 version, these data are preprocessed to reduce dataset bias, with the bias reduction method referring to the scheme proposed by Bai et al.[13]. The specific steps are follows: (1) Drug-target pairs are divided according to the following rules: IC50 < 100 nM indicates positive samples with interaction, IC50 > 10000 nM indicates negative samples without interaction. We use 100-fold difference to reduce label errors, the IC50 threshold is determined according to the work of Gao et al.[14] and Wang et al.[15]; (2) For certain drug, drug-target pairs with only one type of positive or negative sample are deleted to improve the balance of drug-target pairs and reduce hidden biases so that the results are more based on the drug's own feature. The BingdingDB dataset finally used in the experiment contains 14643 drugs and 2623 targets’ interactions. BioSNAP is a balanced dataset created by Huang et al.[16] and Liu et al.[7] from Drugbank database, where the number of positive and negative samples is the same, containing 4510 drugs and 2181 targets’ interactions.

**Baseline model.** *The baseline model used for the regression task.*(1) DeepDTA[10]: A classical model with two branches using CNN modules to extract drug and target features, respectively, and inputting them into a fully connected layer after concatenating the feature vectors of both; (2) CPInformer[3]: Reduces prediction errors caused by similar structural drugs by extracting different drug features, including molecular graph features and fingerprint information of functional groups; (3) GraphDTA[17]: Uses GNN to encode the molecular graph of drugs and CNN to encode the amino acid sequence, modeling DTI. The learned drug and target representations are combined by simple concatenation; (4) AttentionDTA[18]: Uses a joint attention module to generate an attention matrix that measures the binding strength between drug subsequences and each segment of the amino acid sequence; (5) MFR-DTA[19]: Uses multi-layer perceptron (MLP) and CNN modules to extract amino acid sequence features, extracting DTI information through a mixed decoding module, and predicts the corresponding binding regions.

*The baseline model used for the classification task.*(1) Two classical machine learning methods, Support Vector Machine (SVM)[20] and Random Forest (RF)[21], with extended-connectivity fingerprints (ECFPs) and pseudo acid composition (PSC) as input; (2) GraphDTA[17]: To adapt GraphDTA from its original regression task to a binary classification task, a Sigmoid function was added in the last fully connected layer following the steps in previous literature, and then its parameters were optimized by using cross-entropy loss; (3) MolTrans[16]: A model that applies adaptive Transformer layer structures to encode drug and target features; (4) DrugBAN[4]: Uses deep bilinear attention network framework with domain adaptation to explicitly learn pairwise local interaction information between drugs and targets; (5) DeepConv-DTI[22]: Uses and global max-pooling layers to extract local features from amino acid sequences, and fully connected networks to encode drug ECFP fingerprint features. The above DeepDTI models were used with the recommended model hyperparameter settings as described in their original papers.

**Drug feature extractor.** *Input molecular graph.*SAB uses molecular graphs as the input for drugs and employs a deep GCN-based drug feature extractor to extract substructure features. The process of substructure features by deep GCN is shown in Figure 2. As the number of layers increases, the range of molecular features extracted gradually expands.

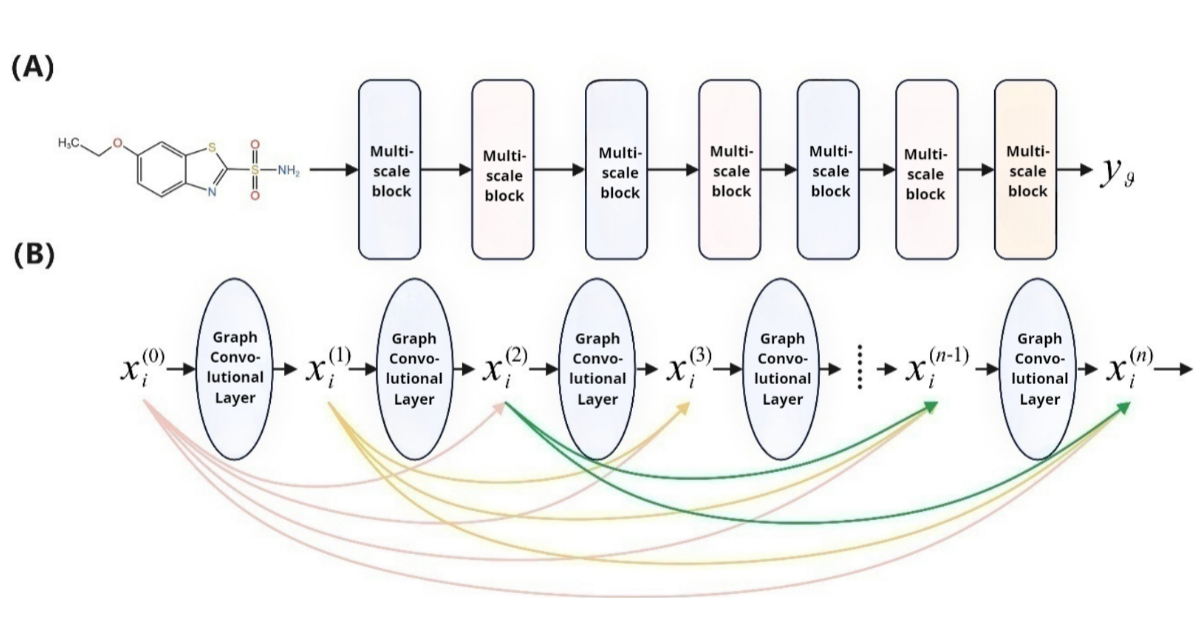


**Figure 2.** Deep GCN extracts drug substructure features

During the construction of molecular graphs, a character-integer pair dictionary for the drug SMILES sequences was firstly constructed, assigning a unique integer to all possible different characters in the SMILES sequences (e.g., various atomic symbols, numbers, special symbols, etc.). This mapping relationship is stored in a dictionary, where each entry contains a character and the corresponding integer value. If the character is not in the dictionary, a value is obtained randomly. The maximum length of the SMILES sequence is set to 100. If the length of the SMILES sequence is greater than 100, the sequence is truncated; if the length is less than 100, the sequence is padded with 0 at the end. It is to ensure that all sequences have the same length before being input into the model, facilitating model processing.

After the above processing, the SMILES sequence is converted into a fixed-length sequence composed of integers. Then, this sequence can be further encoded to a molecular graph through a high-dimensional word vector (which can be regarded as an embedding representation). This molecular graph can capture structural information of the molecule, such as connectivity between atoms, providing input for subsequent graph neural network processing.

*Multi-scale block.* In order to fully extract the substructure features of drugs, the model applies deep GCN to build a drug feature extractor, which consists of three multi-scale blocks and three transition layers, with a transition layer following each multi-scale block. The input to this module is drug molecular graph. The overall structure of drug feature extractor is shown in Figure 3-(A), and the structure of the multi-scale block is in Figure 3-(B).



**Figure 3.** Drug feature extractor based on deep GCN Overall architecture of the drug feature extractor; (B) Structure of the multi block

For each node in the figure, the update formula for the next step is shown as follows:

Here, and are the learned weight matrices shared by all vertices, is the set of adjacent nodes of the vertex i, is batch normalization at the node and also a ReLU activation function. In the next step, the updated node features are fed into the multi-scale block for iteration, to extract more information on the graph. Each multi-scale block contains N graph convolution layers, which introduces dense connections into the GCN, connecting each layer with other layers in a feed-forward manner, allowing all layers to directly access the gradients of the loss function with respect to each weight, thus avoiding the of gradient disappearance and ensuring the performance of deep GCN. The multi-scale block is shown in the following:

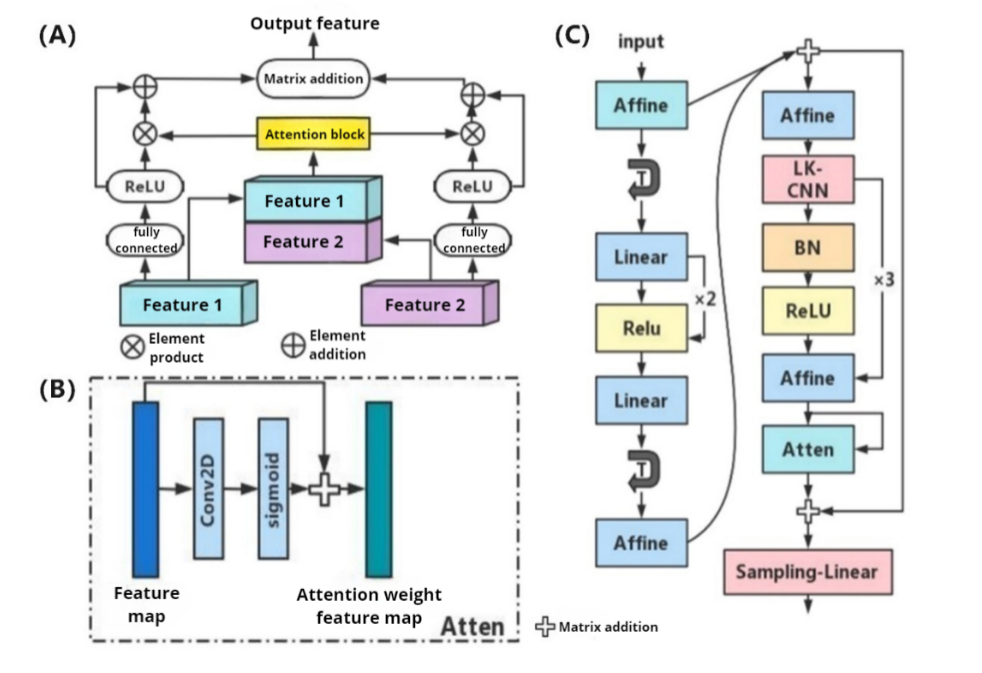
where describes the graph convolutional layer, the parameters are composed of and . represents the th layer, and represents the concatenation operation.The multi-scale block finally outputs multi-scale features including local features and global structural information of the molecule.

*Transition layer.*In order to increase the depth of the GCN, a transition layer is used to connect two adjacent multi-scale blocks, which is used to integrate multi-scale features of the previous multi-scale block, and reduces the number of feature channels. Taking an input multi-scale feature with a stride of N+1 as example, the transition layer is shown in formula (4):

where, and are learnable weight matrices for all nodes. By using a transition layer, the number of channels is reduced to of the input, thus saving computational costs.

*Output drug characteristics.*Finally, the entire molecular graph is converted into a feature vector using the output layer to obtain the final drug features, where is the of vertices in the molecular graph and is the step size at termination. The formula is as follows:

**Multi-scale amino acid encoder.** *Amino acid sequence representation.*SAB takes the amino acid sequence as the target input and uses a multi-scale amino acid encoder based on CNN to extract the substructure features the target, as shown in Figure 4.



**Figure 4.** Multiscale amino acid encoder architecture (A) Target feature fusion module; (B) Attention layer structure in the basic feature extraction; (C) Basic feature extraction module, including global feature extractor (left) and local feature extractor (right)

In the processing of amino acid sequences, similar to the processing of drug SMILES sequences, the fasta format (FASTA) sequences[23] of target proteins are regarded as a language, and employs the methods of Natural Language Processing (NLP) to obtain vector representations. The FASTA sequences are composed of different letters representing different amino acids, which includes:

Each letter represents one of the 22 amino acids, like A represents alanine, and C represents cysteine. There are three additional symbols: B, X, and Z. B and Z are used when aspartic acid and glutamine cannot be distinguished, while X represents an unknown amino acid.

*Basic feature extraction module.*Multi-scale amino acid encoder comprises a basic feature extraction module and a target feature fusion module. Basic feature extraction module is used to convert the amino acid sequence of the target protein into a numerical vector that can be further processed by the model, i.e., the embedding vector. This is somewhat similar to how text is converted into word vectors. The module includes two feature extractors: global feature extractor and local feature extractor. Both extractors use Convolutional Neural Network (CNN). The global feature extractor aims to capture global information across the entire amino acid sequence, such as overall trends and patterns. The local feature extractor focuses on capturing information from specific regions or local patterns within the sequence. To convert the amino acid sequence into an embedding vector, two encoding methods are used: Amino Acid Embedding (AAE) and Word Embedding (WE). These two encoding methods help the model understand each amino acid in the sequence and convert into a numerical form that can be used for mathematical operations.

The specific principle of the basic feature extraction module is as follows: Figure 4-(C) shows the basic feature extraction module for the target, is composed of a global feature extractor and a local feature extractor. These two modules are designed by improving ResMLP[24].

First, the global feature extractor extracts the correlations of different amino acid sequences. This module is composed of an affine block, three fully connected layers and two ReLU layers. The affine layer is used for affine transformation, which includes a linear transformation and a translation. These correspond to the weighted and biased operations of neural networks, respectively, and the formula is as follows:

where is the input for global features, is output, where is the channel size, is the length of target features, represents affine block, which creates a diagonal matrix through function, and are trainable weighted vectors, and consists of a fully connected layer and a ReLU layer.

Based on the global features output from the previous layer, the local feature extractor is used to further mine the local features of the target. In addition to the affine layer and activation function, the local feature extractor adds an extra Local Kernel Convolutional Neural Network (LK-CNN) layer and an attention layer. The LK-CNN layer uses more convolutional structures, including three convolutional blocks. Each convolutional block consists of a one-dimensional convolutional layer with a large kernel, a Batch Normalization (BN) layer, and a ReLU activation function layer, which can more effectively extract complex biological sequences such as amino acid sequences. The formula for the local feature extractor is as follows:

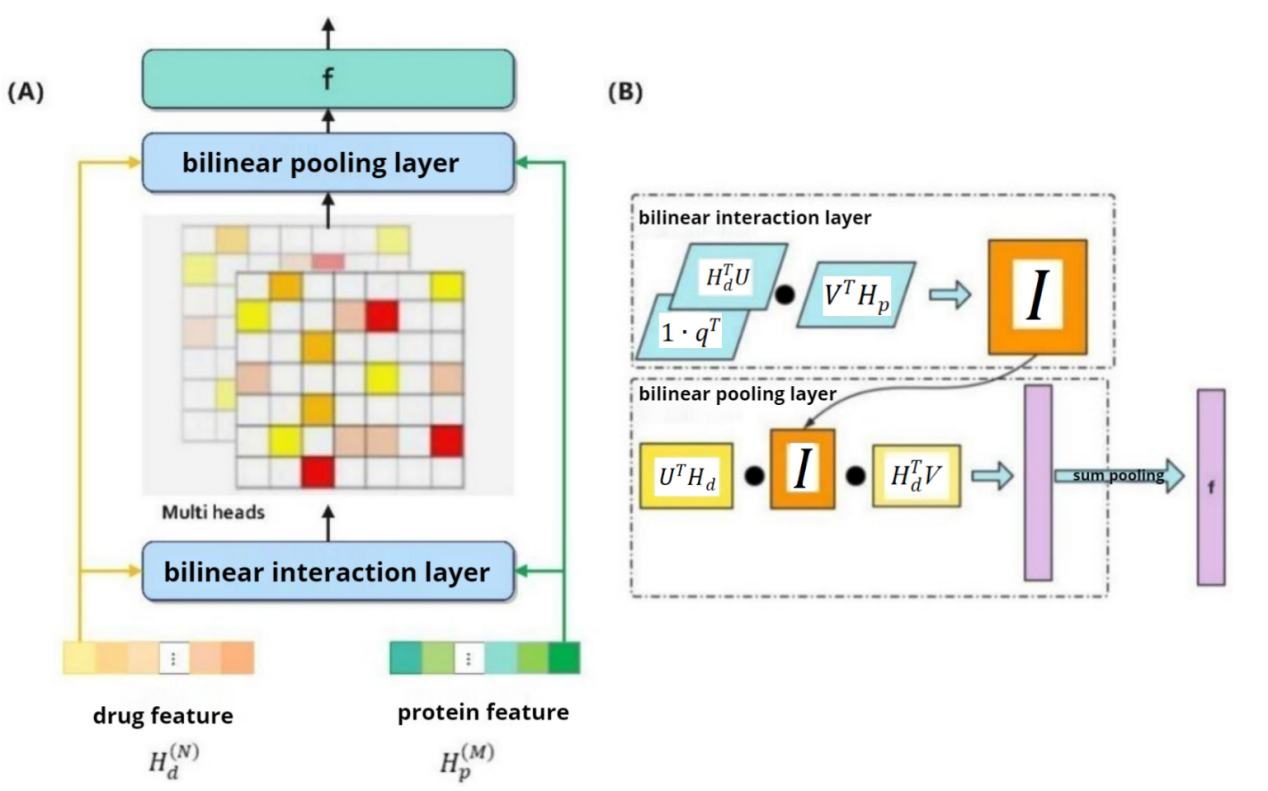
Among them, is output features of the local feature extractor, where are Target characteristic length and channel size, is fully connected layer, is LK-CNN layer.

In addition, Att(X) is the spatial attention module, which captures local relationships between adjacent elements through 2D-convolution and normalizes captured information through Sigmoid() function to enrich individual features. Figure 4-(B) shows the structure of the attention layer, and the formula is as follows:

*Target Feature Fusion Module.*The basic feature extraction module completes the encoding and feature extraction of the amino acid sequence, and the next is to integrate these features to represent the target protein more comprehensively. This is the task of the target feature fusion module. The target feature fusion module mainly consists a fully connected layer (also known as a dense layer) and a ReLU activation function. The fully connected layer can learn how to combine information from different feature extractors. Two types of target features extracted by the basic feature extraction module are input into the feature fusion block for feature fusion. The ReLU activation function is applied the output of the fully connected layer to introduce non-linearity. This allows the model to capture more complex patterns and relationships. After processing by the feature fusion module we obtain the final target features[19], which integrate information from global and local feature extractors, providing a rich representation for subsequent analysis or prediction tasks. Structure of the target feature fusion module is shown in Figure 4-(A), where global features and local features are combined through an addition operation, and then through fully connected layer, resulting features are fed into the feature fusion block. The formula for the target feature fusion block is as follows:

Where consists of a fully connected layer and an activation layer ReLU, is the result of and concatenated and input into for 2D-convolution operations. The Sigmoid function is used to normalize the features, resulting in the attention weight matrix for feature and (1-) is the attention weight matrix for feature . \* indicates element-wise multiplication. Finally, the two features are combined to obtain the final target feature.

**Feature Fusion Module Based on bilinear Attention.** *Overall structure.*We have already obtained the drug features and target features, but still need to combine and compute their interactions in order to predict the drug-target affinity. Figure 5 shows the structure of the feature fusion module based on bilinear attention: By using the bilinear interaction layer and the bilinear pooling layer, the attention weights and interaction features between drug and target features are calculated respectively, so as to extract the key information between drug-target substructures and capture the complex interactions between drugs and targets.



**Figure 5.** Structure of the feature fusion module based on bilinear attention (A) Overall structure of the feature fusion module; (B) Formula the bilinear attention mechanism

The Bilinear Attention Network (BAN) is an attention-based model that was first introduced in the field of Visual Question Answering (VQA) in computer vision. Given an image and a related natural language question, the goal of VQA is to predict a text-image matched answer[25]. Thus, VQA can be seen as a multi-modal learning task with application scenarios similar to the Drug-Target Interaction (DTI) prediction task. BAN extends the linear attention network with bilinear attention maps to adapt to multi-modal learning. This network structure considers the information of each pair of multi-modal input channels i.e., the interaction representation of a region in the image and the corresponding question word. At the same scale, compared to directly using multi-modal data, BAN with one attention mechanism can provide richer joint information at a lower computational cost. Due to the similarity between VQA and Drug-Target Affinity (DTA), inspired by the work of Bai et al.[4], a drug-target interaction module was built by BAN to capture the pairwise interactions between drugs and targets, as shown in Figure 5-(A).

The input to the module are drug features and target features, and the structure of the module consists of two bilinear layers: (1) a bilinear interaction layer for capturing pairwise attention weights; (2) a bilinear pooling layer for extracting the connections between drugs and targets.

*Bilinear Interaction Layer.* The formula for the bilinear interaction layer is as follows:

where 、 are learnable weight matrices for drug and target representations, is a learnable weight vector, and 1 is a fixed all-one vector. “。” denotes the Hadamard product. Elements in I represent the interaction strength between drug-target substructure pairs and are mapped to latent binding sites and molecular substructures, as shown in formula (12). Here, represents the i-th substructure representation of the drug, and represents the j-th substructure representation of the target. Their binding strength is denoted as , and and are mapped to a common feature space with weight matrices of and , learning the Hadamard product and the weights of the vector .

*Bilinear Pooling Layer.*The formulas for the bilinear interaction layer and pooling layer are shown in Figure 5-(B), where the output of the bilinear interaction layer will continue to participate in the computation of the pooling layer. In order to obtain the joint representation of drug and target, a bilinear pooling layer is built on the interaction map , and the formula for the k-th layer of the bilinear pooling layer is as follows:

where and denote the k-th column of the weight matrices and , respectively.

*Output prediction results.* Finally, add a pooling layer to sum up the multiple features, resulting in the final mapping:

The final joint features firstly pass through a three-layer fully connected layer, with ReLU activation and BN layers added, and finally through fully connected layer to obtain the SAB prediction results.

**Implementation steps.** *Target characteristic input.*We first input the target features to the multi-scale amino acid encoder. The input of the target is an ASCII string representing the amino acids. The gene names of the targets were obtained from the UniProt[26] database, with each character representing an amino acid residue. Each amino acid is encoded to a integer based on its alphabetical symbol, e.g., Alanine (A) = 1, Cysteine (C) = 3, Aspartic acid (D) = 4, etc. The sequence of the target protein "AACGFED" be represented as [1126543]. We use WE[3] and AAE for the input features of the target. WE treats the sequence of the target protein as a sentence, with each character in the sentence as a word. The integer-encoded sequence is used as the input for the embedding layer, using the "one-hot" encoding rule. The output of the embedding layer is a 128-dimensional vector representation, saved as a pt file. Due to the different lengths of the data, to reduce computational complexity while retaining enough valid information, a fixed length is set to obtain valid representations, which is set to 1000 in the experiment. That is, sequences longer than 1000 are truncated, and shorter ones are padded with zeros. Amino acid sequences are typically an ordered arrangement of the 20 amino acids shared by the primary structure. If the sequence is treated as a text and three amino acid fragments are treated as a word, there will be 10,648 (223) different words in the generated vocabulary. The information contained may be far richer than the meaning of the sequence itself. In this case, the extracted target features are too sparse, so AAE will used. Following the methods of Chen et al.[27], 22 amino acids are divided into 6 classes based on their biochemical properties:, , , , , . The number of words is compressed to 216 (63), reducing the redundancy of protein features. For example, the sequence 'MSPLNQSAEGLPQEASNRSLN' can be converted to 'EDDEBBDDBDEDBBDDBADEB'.

*Drug feature input.*First, the sequence data of drugs collected from the database is in the form of SMILES, a simplified notation for chemical molecular formulas. It transforms the chemical structure of molecules into one-dimensional strings through ASCII characters, which is very convenient for various molecular editors to read. The next important step is to use RDKit 2021.03.2[2] software package based on Python to convert the SMILES strings into matrix representations of molecular (spatial topological structures). RDKit is an open-source toolkit for cheminformatics, used for fingerprint generation, descriptor generation of drugs, etc.. SMILES are encoded as molecular graphs G = (V, E) via RDKit and DeepChem[28], where V represents set of atoms in a drug, with each atom represented by a feature vector (F[i], V ∈ {i1, i2...in}) which stores all atom features within each molecular graph as an atom feature matrix F. This feature vector is composed of five features, namely atom type, atom degree (the sum of neighbor and hydrogen’ numbers bonded to the atom), total number of hydrogen, atomic implicit value, and whether the atom is aromatic. Algorithm 1 shows the specific operations of the algorithm for converting SMILES to molecular graphs. E is represented as the bonds of a drug that indicates the dependency relationships between each of atoms, and is stored in an adjacency matrix A(U[i], V[i]). These structural descriptors are encoded using the “one-hot” encoding rule, and encoded structural information is used as input for feature extraction in neural networks. The computation process is shown in formula 16, where the one-hot encoding for atoms calculated as follows:

**Algorithm 1. Converting SMILES into molecular graphs**

|  |
| --- |
| **Input:** SMILES sequence S  **Output:** Atom count C, set of atom feature vectors F, and adjacency matrix E of undirected graph |
| Start  Based on S, get the corresponding chemical structure mol through the MolFromSmiles() function;  Get the value of C through the function GetNumAtoms() based on mol;  Based on mol, get the set of atoms = [, ,…, ] through the function Get Atoms();  For each atom in do  Though the function GetSymbol(), get the atomic symbol of ;  Though the function GetDegree(), get the atomicity of ;  Though the function GetTotalNumHs(), get the total hydrogen content of ;  Though the function GetImplicitValence()，get Atomic implicit value of ;  if is aromatic do  get = True;  else  get = False;  end if  for in , do  if not an integer between 0 and 10 then  Warning: is not in the permitted dataset  else if not an integer between 0 and 10 then  Append the value of to the end of the authorized dataset；  end if  Based on the licensed dataset, represent as a one-hot vector ;  end for |
| These 5 one-hot vectors (k=1,...,5) are concatenated into a vector , which is then calculated according to formula (16) to obtain ;  end for  Though the function GetBonds(), based on get the edge set , ,…, ;  for each edge in do  Though the function GetEndAtomIdx()，Get the starting atom and the ending atom of edge |

After the drug features and target features are input, the software will start the experiment. The experiments of SAB were conducted on a server with an AMD EPYC 7601 CPU and an NVIDIA GeForce RTX 3070 8G card, implemented in Python3.8[29] with PyTorch 1.12.1[30] environment, using functions from RDKit 2021.03.2[31], Numpy1.23.5[32], scikit-learn1.2.2[33], Pandas1.4.1[34], torch-geometric 2.3.0[35].

After preparing the data and designing the model, the next steps of forward propagation, loss computation, back propagation, and weight update are the core steps in neural network training process.

*Forward propagation, compute loss and back propagation.*During the training process of each iteration, when the network performs forward propagation, for each input sample, the network will randomly "drops" (i.e., set to zero) the output of some neurons. This drop process is carried out according to a predetermined dropout rate. This is because compared with classical machine learning models, deep learning models are more prone to overfitting and underfitting risks during training. Underfitting is often manifested as not fitting data well on the training set, while overfitting is manifested as good performance on the training set but poor performance on the validation and test sets[36]. Regularization is a method to avoid overfitting and improve generalization ability. Common regularization techniques mainly include L1 regularization, L2 regularization, early stopping, dropout. Therefore, when training the SAB model, we employed the dropout technique to avoid overfitting, and used the network output after dropout to calculate loss function value. In regression tasks, the mean squared error (MSE) is used as the model's loss function, with inputs being the predicted results and corresponding labels. All learnable parameters are jointly optimized through back propagation. In classification tasks, the classical binary cross entropy (BCE) is used as the model's loss function:

Where refers to the prediction and refers to the label. If a drug interacts with the corresponding target, the label 1 is assigned the drug-target pair; if a drug does not interact with the corresponding target, the label 0 is assigned to the drug-target pair. The gradient is calculated on the loss function and backpropagation is performed. During back propagation, only the neurons that were retained (not dropped) during the forward propagation participate in gradient computation. The gradient is the derivative of a multivariate function at a point, pointing in the direction of the fastest increase of the loss function.

*Weight update.*During the SAB model training, the adaptive moment estimation (Adam) optimizer was used to perform a gradient optimization, which is reflected in the weight update. In deep learning, we usually use the gradient to refer to the derivative of the loss function with respect to the model parameters. The direction of the gradient vector points to the direction of the fastest growth of the loss function. Therefore, in order to minimize the loss function, we need to update the parameters in the opposite direction of the gradient. Dropout occurs again after this step, that is, it is applied again in the next forward propagation. Unlike the stochastic gradient optimizer, the Adam optimizer can flexibly adjust the learning rate to optimize the model. When training the model, it is necessary to minimize the loss function between the labels and the predicted results. The modeling is divided into regression tasks and classification tasks. The regression task predicts the specific binding affinity value, reflecting the of the interaction between the drug and the target; the classification task predicts whether there is an interaction between the drug and the target.

*Hyperparameter selection.*Different hyperparameters can also lead to different computational results. For deep learning algorithms, hyperparameters mainly include regularization coefficients, network architecture (e.g., the number of neurons per layer, the number of network layers, the activation function), and optimization-related hyperparameters (e.g., learning rate and batch size). Hyperparameters are different from learnable parameters; they are used to define the model architecture and model strategy. The purpose of hyperparameter search is to find the optimal values of these parameters to improve the model's performance. Common hyperparameter optimization methods mainly include Bayesian, grid search, random search, and neural architecture search. According to experience, grid search is a simple and practical hyperparameter search method, and SAB uses grid search to optimize the model. Its basic idea is to traverse the given hyperparameter grid, trying all possible combinations of hyperparameters, then selecting the best one based on the performance on the validation set. In addition, the learning rate, the number of neurons per layer, and the dropout rate often have significant impact on the model's performance. The hyperparameter search range is shown in Table 3, with the ReLU activation function fixed, and the search was on three sets of parameters: learning rate, the number of neurons per layer, and the dropout rate. The model with the optimal MSE or AUROC on the set was saved and used to evaluate the final performance on the test set. The final selection was a batch size of 32, a learning rate of ×10-5, a dropout rate of 0.2, and 100 training layers.

**Table 3. Hyperparameter settings of the SAB model**

|  |  |
| --- | --- |
| Hyperparameter | Range of values |
| Batch size | 32  64  128 |
| Learning rate | 1×10-2  1×10-3  1×10-4  5×10-5 |
| Dropout | 0.1  0.2  0.3 |

*Model evaluation metrics.*In order to measure the performance of the DTI model, it is necessary to make predictions for each sample in the training set, validation set and test set using the model, and then calculate the evaluation metrics based on the model's predicted results and the given labels. For regression tasks, the model's is evaluated using the consistency index (CI), mean squared error (MSE), and r2m. The higher the CI and r2m are, the lower MSE is, and the better the model's predictive performance.

The formula for calculating the CI index is shown in formula (18):

where > ， is the predicted value of the i-th sample， is the predicted value of the j-th sample， is a normalization constant， is a step function：

MSE is commonly used as the L2 loss function for linear regression models. When used for model evaluation, MSE straightforwardly expresses the prediction error. However, the squaring amplifies the error values, leading to unclear evaluation results when the magnitude is very large. When used as a loss function for training, the model is greatly affected by outliers, as shown in formula (19), where is the true value and represents the predicted value.

The r2m index is used to measure the external predictive performance of the model, and when the r2m value of the test set >0.5, it indicates that the model is valid. The calculation formula is shown in formula (20) :

where r2 and r20 are the squares of the correlation coefficients between observed and predicted values with and without intercept, respectively.

The performance of the classification task model was evaluated using five metrics: area under the receiver operating characteristic curve (AUROC), area under precision-recall curve (AUPRC), accuracy, sensitivity, and specificity. These evaluation metrics were calculated using a self-written Python script, as detailed in Table 4.

**Table 4. Model evaluation metrics**

|  |  |  |
| --- | --- | --- |
| Task type | Metrics | Introduction |
| Regression task | CI |  |
|  | MSE |  |
|  | r2m |  |
| Classification task | AUROC | area under receiver operating characteristic curve |
|  | AUPRC | area under precision-recall curve |
|  | Accuracy |  |
|  | Sensitivity (Recall) |  |
|  | Specificity |  |

Note: TP refers to the number of true positives, FN refers to the number of false negatives, TN refers to the number of true negatives, FP refers to the number of false positives. True positive refers to a positive prediction that is actually positive; true negative refers to a negative prediction that is actually; false positive refers to a positive prediction that is actually negative; false negative refers to a negative prediction that is actually positive.

## 

1. **Results**

**Regression task results.** For the regression tasks on the Davis and KIBA datasets, the model performance was evaluated using CI and MSE, where a larger CI value and lower MSE value indicate better model prediction performance, and was compared with the DeepDTA, CPInformer, GraphDTA, AttentionDTA, and FR-DTA models.

The performance evaluation results of the SAB model and the baseline models on the Davis and KIBA datasets are shown in Table 5. According to the results, in the Davis dataset, the imbalanced label distribution leads to most models predicting affinity towards a smaller value, affecting the metrics. But the SAB model still performs well, with a CI value of 0.881, an MSE value of 0.208, a value of 0.660. The CI and values do not exceed those of the current SOTA model MFR-DTA, but the MSE shows a performance improvement of 0.8%. In the KIBA dataset, the label distribution is relatively normal, but most of the samples in the dataset highly concentrated, making it difficult to predict the trend of affinity. The SAB model does not show the best performance, with a CI value of 0.876, an MSE value of 0.155, and a value of 0.686. Overall, the experimental results show that is no particularly outstanding performance on these two datasets, with only some excellent performance in certain evaluation metrics.。

**Table 5. Performance evaluation results in the regression task (Davis, KIBA datasets)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dataset | Methods | CI↑ | MSE↓ | ↑ |
| Davis | DeepDTA | 0.878 (0.004) | 0.261 | 0.630 (0.017) |
| CPInformer | 0.874 (0.002) | 0.277 | 0.618 (0.004) |
| GraphDTA | 0.890 (0.005) | 0.233 | 0.663 (0.010) |
| AttentionDTA | 0.893 (0.005) | 0.216 | 0.677 (0.014) |
| MFR-DTA | 0.905 (0.001) | 0.221 | 0.705 (0.003) |
| **SAB** | 0.881 (0.002) | **0.208** | 0.660 (0.003) |
| KIBA | DeepDTA | 0.863 (0.002) | 0.194 | 0.673 (0.009) |
| CPInformer | 0.867 (0.003) | 0.183 | 0.677 (0.002) |
| GraphDTA | 0.883 (0.004) | 0.151 | 0.687 (0.010) |
| AttentionDTA | 0.882 (0.004) | 0.155 | 0.755 (0.017) |
| MFR-DTA | 0.898 (0.002) | 0.136 | 0.789 (0.002) |
| **SAB** | 0.876 (0.003) | 0.155 | 0.686 (0.008) |

**Classification task results.** For the classification tasks on the BindingDB and BioSNAP datasets, we evaluate the models using AUROC, AUPRC, accuracy, sensitivity, and specificity. Each dataset was randomly split into training, validation, and test sets in an 8:1:1 ratio. For each dataset, we perform three independent runs using random seeds, selected the model with the best AUROC on the validation set as the best model, and then evaluated the selected model on the test set.

SAB is compared with the following six baseline models: SVM, RF, DeepConv-DTI, GraphDTA, MolTrans DrugBAN, and the evaluation results of the models on the BindingDB and BioSNAP datasets are shown in Table 6. In the BindingDB dataset, SAB results in AUROC = 0.962, AUPRC = 0.944, Accuracy = 0.92, Sensitivity = 0.904, Specificity = 0.915, all of which are better than the baseline models except that AUPRC is slightly lower than DrugBAN, but still competitive. In the BioSNAP dataset, SAB results in AUROC = 0.909, AUPRC = 0.908, Accuracy = 0.852 Sensitivity = 0.857, Specificity = 0.842, with superior AUROC, AUPRC, and sensitivity compared the baseline models, and also good accuracy and specificity results.

**Table 6. Performance evaluation results of the SAB model in the classification task (BindingDB, BioSNAP datasets)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Dataset | Method | AUROC | AUPRC | Accuracy | Sensitivity | Specificity |
| BindingDB | SVM | 0.939(0.001) | 0.928(0.002) | 0.825(0.004) | 0.781(0.014) | 0.886(0.012) |
| RF | 0.942(0.011) | 0.921(0.016) | 0.880(0.012) | 0.875(0.023) | 0.892(0.020) |
| DeepConv-DTI | 0.945(0.002) | 0.925(0.005) | 0.882(0.007) | 0.873(0.018) | 0.894(0.009) |
| GraphDTA | 0.951(0.002) | 0.934(0.002) | 0.888(0.005) | 0.882(0.012) | 0.897(0.008) |
| MolTrans | 0.952(0.002) | 0.936(0.001) | 0.887(0.006) | 0.877(0.016) | 0.902(0.009) |
| DrugBAN | 0.960(0.001) | 0.948(0.002) | 0.904(0.004) | 0.900(0.008) | 0.908(0.009) |
| **SAB** | **0.962(0.002**) | 0.944(0.001) | **0.923(0.001**) | **0.904(0.004**) | **0.915(0.008**) |
| BioSNAP | SVM | 0.862(0.007) | 0.864(0.004) | 0.777(0.011) | 0.711(0.042) | 0.841(0.028) |
| RF | 0.860(0.005) | 0.886(0.005) | 0.804(0.005) | 0.823(0.032) | 0.786(0.025) |
| DeepConv-DTI | 0.886(0.006) | 0.890(0.006) | 0.805(0.009) | 0.760(0.029) | 0.851(0.013) |
| GraphDTA | 0.887(0.008) | 0.890(0.007) | 0.890(0.007) | 0.745(0.032) | 0.854(0.025) |
| MolTrans | 0.895(0.004) | 0.897(0.005) | 0.897(0.005) | 0.818(0.031) | 0.831(0.015) |
| DrugBAN | 0.903(0.004) | 0.902(0.004) | 0.902(0.004) | 0.820(0.021) | 0.847(0.010) |
| **SAB** | **0.909(0.005**) | **0.908(0.006**) | 0.852(0.004) | **0.857(0.015**) | 0.842(0.010) |

**Cold start scenario evaluation.** However, using random splits to divide the training, validation, and test sets may cause information about the drugs and targets leaking from the training set the test set, leading to overly optimistic results. Therefore, in practical application scenarios, the model needs to extrapolate to predict interactions for drugs, targets, and-target pairs that are not seen in the training set.

Therefore, the performance of the model was evaluated using three new splitting methods in the cold start scenario: drug cold start, target cold start and drug-target pair cold start. In machine learning and recommendation systems, the cold start problem refers to the situation where the model needs to make predictions for new users or new items without historical data. For example, in the case of drug cold start, drugs were divided into training set, validation set, and test set, with in the test set not appearing in the training and validation sets, and drugs in the training and validation sets being different as well. Similarly, target cold start was on different targets. These three splitting methods can better demonstrate the model's generalization ability and meet the needs of the new drug discovery process.

SAB employs the Davis dataset and the BindingDB dataset for the cold-start studies, including the drug cold start, the target cold start, and the double cold tasks. Specifically, under the drug cold start scenario, the Davis dataset was split into 54, 7, and 7-overlapping drugs. The BindingDB dataset was split into 11698, 1456, and 1464 non-over drugs. Under the target cold start scenario, the Davis dataset was split into 354, 44, and 44 non-overlapping targets and the BindingDB dataset was split into 2116, 239, and 268 non-overlapping targets. As for the cold tasks, since the BindingDB dataset has a biased split, this study only employs the Davis dataset for the double cold task. The detailed data statistics are shown Table 7:

**Table 7. The division of the cold start dataset**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| dataset | tash | train | valid | test |
| Davis  BindingDB | Cold target  Cold drug  All cold  Cold target  Cold drug  All cold | 24072(354)  23868(54)  19116  39159 (2116)  39441(11698)  31993 | 2992(44)  3094(7)  308  4735(239)  5001 (1456)  395 | 2992(44)  3094(7)  308  5305 (268)  4757(1464)  509 |

The study conducted drug cold start, target cold start, and double cold tasks on the Davis dataset. The results are shown in Table 8, SAB achieved MSE=0.650 and CI=0.720 in the drug cold start task, MSE=0.47 and CI=0.766 in the target cold start task, and MSE=0.883 and CI=0.632 in the double cold task. From the comprehensive analysis of the results for the three cold start tasks, SAB model performs slightly better than GraphDTA on the Davis dataset but neither of them surpasses the current SOTA model NHGNN.

**Table 8. Performance evaluation results of SAB in cold start scenarios (Davis dataset)**

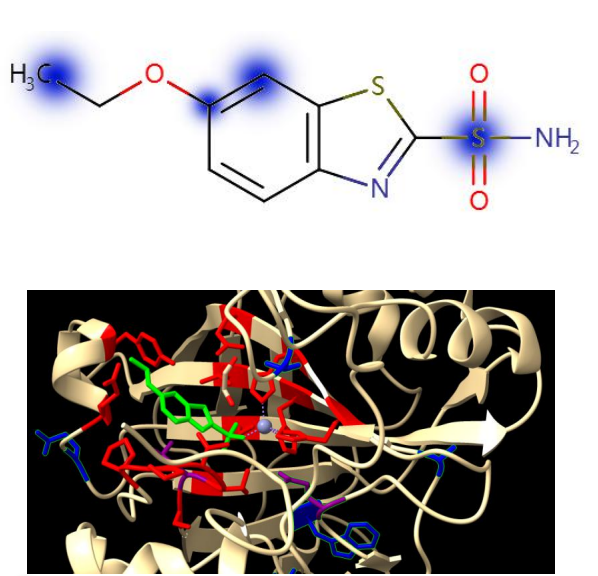
|  |  |  |  |
| --- | --- | --- | --- |
| scene | model | MSE↓ | CI↑ |
| Cold drug | GraphDTA | 0.920 (0.029) | 0.678(0.036) |
|  | MgraphDTA | 0.563 (0.065) | 0.729(0.022) |
|  | FusionDTA | 0.581 (0.094) | 0.737(0.012) |
|  | NHGNN | 0.554 (0.091) | 0.752(0.017) |
|  | **SAB** | **0.650 (0.003)** | **0.720(0.015)** |
| Cold target | GraphDTA | 0.510(0.086) | 0.729(0.012) |
|  | MgraphDTA | 0.359(0.023) | 0.813(0.008) |
|  | FusionDTA | 0.364(0.021) | 0.826(0.011) |
|  | NHGNN | 0.344(0.029) | 0.855(0.016) |
|  | **SAB** | **0.472(0.004)** | **0.766(0.002)** |
| All cold | GraphDTA | 0.968(0.006) | 0.579(0.017) |
|  | MgraphDTA | 0.874(0.030) | 0.636(0.021) |
|  | FusionDTA | 0.876(0.021) | 0.645(0.043) |
|  | NHGNN | 0.857(0.016) | 0.665(0.038) |
|  | **SAB** | **0.883(0.012)** | **0.632(0.004)** |

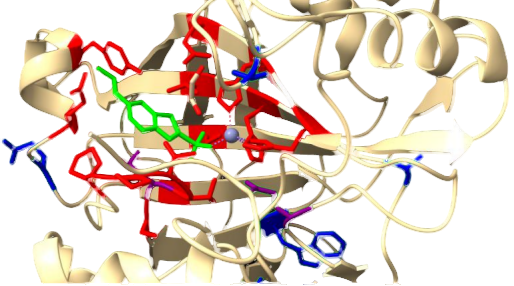
The research was conducted on the BindingDB dataset for drug cold start and target cold start tasks, as shown in Table 9. In drug cold start task, the result of SAB was AUROC=0.877, and in the target cold start task, the result of SAB was AUROC=0.569, which did not exceed the SOTA model DrugBAN and was even worse than all baseline models. Therefore, SAB's overall performance on the Davis dataset and the BindingDB dataset was poor, and the model still has a lot of room for improvement.

**Table 9. Performance evaluation results of SAB cold start scenario (BindingDB dataset)**

|  |  |  |
| --- | --- | --- |
| scene | model | AUROC↑ |
| Cold drug | DeepConv-DTI | 0.943 (0.004) |
|  | GraphDTA | 0.950 (0.004) |
|  | MolTrans | 0.945 (0.004) |
|  | DrugBAN | 0.959 (0.002) |
|  | **SAB** | **0.877 (0.003)** |
| Cold Target | DeepConv-DTI | 0.627 (0.070) |
|  | GraphDTA | 0.670 (0.023) |
|  | MolTrans | 0.661 (0.037) |
|  | DrugBAN | 0.692 (0.038) |
|  | **SAB** | **0.569 (0.014)** |

**Result visualization.** Meanwhile, we can visualize the results using a dual-attention network architecture. In the dual-attention network, through the attention mechanism the model can assign a weight to each substructure of the drug and target, which reflects the importance of that substructure to the final prediction result. After the training is completed, researchers can analyze these attention weights to determine which substructures of the drugs and targets contribute the most to the affinity prediction, and present these weights in a graphical manner, thus intuitively understanding the basis of the model's predictions. When selecting target proteins and drugs, the following conditions are referred to: (1) X-ray structure resolution of human target proteins > 2.5 Å; (2) Co-crystallized ligand pIC50 ≤ 10 nM, and the drug does not appear in the training set. In this study, we use the PDB Bind database's Etazolamide in complex with carbonic anhydrase 2 (PDB ID:6QL2) for case analysis, and the visualization results are shown in Figure6.





**Figure 6.** Visualization of 6QL2 and Ethoxazole amide Predictions

Following the work of Bai et al.[4], for each drug molecule, the top 10% weighted atoms in bilinear attention graph were colored blue in the drug, and the target protein structures associated with the drug were colored red. The top 10 structures in the bilinear attention graph were colored blue, and the overlapping parts were colored purple. Whether there is an overlap was used to judge if the model prediction is valid. In ethoxazole, SAB explained that the sulfonamide region is the main area where the drug binds to the target, with the sulfonamide oxygen acting as hydrogen bond receptor for the main chain of Leu-198 and Thr-199, and the amino group acting as a hydrogen bond receptor for the side chains of His-94 and Thr-199. However, SAB incorrectly predicted specific interactions between other atoms and the targets. In target protein, 7 residues predicted by the model were highlighted: Ala-115, Pro-137, Glu-106 Thr-198, Lys-256, Glu-26, and Thr-65, among them Thr-198 and Gl-106 forming specific interactions with ethoxazole.

The results indicate that there is still room for studying the explainability of the model for drug-target binding sites. The target inputs used in SAB are one-dimensional amino acid sequences, which, although incorporating individual features with global features during feature extraction, do not consider the three-dimensional structure of the target, causing potential significant errors in predicting drug-target binding sites. In future research, the three-dimensional information of the target can be incorporated into the model framework to further improve explainability of the drug-target affinity prediction model.

## **Conclusion and Discussion**

To address the issue of substructure in drug-target interaction modeling, this paper proposes a Substructure-based Graph convolutional drug-target Affinity Bilinear attention Prediction model, SAB. In this model, a deep GCN-based drug feature extractor is used to extract drug substructure features, and a multi-scale amino acid encoder based on CNN is used extract target substructure features. Finally we use a feature fusion module based on bilinear attention mechanism to capture the attention weights of drug-target pairs and extract the interaction features of drug-target substructures.

The SAB model shows a significant advancement in the realm of drug-target affinity prediction, distinguished by its unique focus on the substructures of drugs and targets. This innovative approach, combining GCN with multi-scale amino acid encoders and a dual linear attention mechanism, has demonstrated a robust capability to extract and integrate intricate features, leading to enhanced predictive accuracy and interpretability.

By designing contrastive experiments, the performance of the SAB model on regression tasks and classification tasks was analyzed. Compared with the regression baseline models, the MSE of SAB was slightly better than the current SOTA model MFR-DTA, while the other metrics weren’t. So the overall performance of SAB on the regression task was average. However, the AUROC of SAB model on the classification tasks surpassed multiple baseline models, showing relatively excellent results on the classification task. These results underscore SAB's strength in categorical predictions, which is a critical aspect in drug discovery and development processes.

In addition, in three cold-start tasks, analyzing the results of the two tasks comprehensively, the performance of SAB did not exceed the baseline models, showing poor performance.We found this may be related to the model's insufficient utilization of the three-dimensional spatial structure of drugs and target molecules through visualization. Therefore, we can input the spatial structure into the SAB model, and develop new algorithms to use them.

Compared with traditional DTA methods, the advantages of the SAB model are mainly reflected in the following aspects. Firstly, SAB model focuses on extracting the substructure features of drug-target complex, which helps to predict the affinity between drugs and targets more accurately. Secondly, the bilinear attention mechanism allows the model to recognize and utilize the key interaction information between drug and target substructures, thereby improving the accuracy of predictions. Additionally the SAB model provides visual analysis of drug-target binding sites, enhancing the interpretability of the model and enabling researchers to intuitively understand the model's results. Finally, the performance of the SAB model on multiple datasets demonstrates its generalization ability in different scenarios. These advantages indicate that the SAB model has significant application potential in the fields of drug discovery and drug design.

## **Reference**

1. Yang Z, Zhong W, Zhao L, et al. MGraphDTA: deep multiscale graph neural network for explainable drug–target binding affinity prediction [J]. Chem Sci, 2022, 13(3): 816-833.
2. RDKit: Open-source cheminformatics. Release 2014.03.1 | Semantic Scholar [EB/OL].
3. Hua Y, Song X, Feng Z, et al. CPInformer for Efficient and Robust Compound-Protein Interaction Prediction [J]. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2023, 20(1): 285-296.
4. Bai P, Miljković F, John B, et al. Interpretable bilinear attention network with domain adaptation improves drug–target prediction [J]. Nat Mach Intell, 2023, 5(2): 126-136.
5. Davis MI, Hunt JP, Herrgard S, et al. Comprehensive analysis of kinase inhibitor selectivity [J]. Nat Biotechnol, 2011, 29(11): 1046-1051.
6. Tang J, Szwajda A, Shakyawar S, et al. Making Sense of Large-Scale Kinase Inhibitor Bioactivity Data Sets: A Comparative and Integrative Analysis [J]. J Chem Inf Model, 2014, 54(3): 735-743.
7. Liu H, Sun J, Guan J, et al. Improving compound–protein interaction prediction by building up highly credible negative samples [J]. Bioinformatics, 2015, 31(12): i221-i229.
8. Liu T, Lin Y, Wen X, et al. BindingDB: a web-accessible database of experimentally determined protein-ligand binding affinities [J]. Nucleic Acids Research, 2007, 35(Database): D198-D201.
9. BioSNAP: Network datasets: Drug-drug interaction network [EB/OL]. [2024-02-17].
10. Öztürk H, Özgür A, Ozkirimli E. DeepDTA: deep drug–target binding affinity prediction [J]. Bioinformatics, 2018, 34(17): i821-i829.
11. Pahikkala T, Airola A, Pietila S, et al. Toward more realistic drug-target interaction predictions [J]. Briefings in Bioinformatics, 2015, 16(2): 325-337.
12. He T, Heidemeyer M, Ban F, et al. SimBoost: a read-across approach for predicting drug–target binding affinities using gradient boosting machines [J]. J Cheminform, 2017, 9(1): 1-14.
13. Bai P, Miljković F, Ge Y, et al. Hierarchical Clustering Split for Low-Bias Evaluation of Drug-Target Interaction Prediction [C]. 2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). 2021: 641-644.
14. Gao KY, Fokoue A, Luo H, et al. Interpretable Drug Target Prediction Using Deep Neural Representation [C]. Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence. Stockholm, Sweden: International Joint Conferences on Artificial Intelligence Organization, 2018: 3371-3377.
15. Wang Z, Liang L, Yin Z, et al. Improving chemical similarity ensemble approach in target prediction [J]. Journal of Cheminformatics, 2016, 8(1): 20.
16. Huang K, Xiao C, Glass LM, et al. MolTrans: Molecular Interaction Transformer for drug–target interaction prediction [J]. Bioinformatics, 2021, 37(6): 830-836.
17. Nguyen T, Le H, Quinn TP, et al. GraphDTA: predicting drug–target binding affinity with graph neural networks [J]. Bioinformatics, 2021, 37(8): 1140-1147.
18. Zhao Q, Duan G, Yang M, et al. AttentionDTA: Drug-Target Binding Affinity Prediction by Sequence-Based Deep Learning With Attention Mechanism [J]. IEEE/ACM Trans Comput Biol Bioinform, 2023, 20(2): 852-863.
19. Hua Y, Song X, Feng Z, et al. MFR-DTA: a multi-functional and robust model for predicting drug–target binding affinity and region [J]. Bioinformatics, 2023, 39(2): btad056.
20. Suthaharan S. Support Vector Machine [M]. Machine Learning Models and Algorithms for Big Data Classification. Springer, Boston, MA, 2016: 207-235.
21. Rigatti SJ. Random Forest [J]. Journal of Insurance Medicine, 2017, 47(1): 31-39.
22. Lee I, Keum J, Nam H. DeepConv-DTI: Prediction of drug-target interactions via deep learning with convolution on protein sequences [J]. PLoS Comput Biol, 2019, 15(6): e1007129.
23. Pearson, W. R., & Lipman, D. J. (1988). Improved tools for biological sequence comparison. Proceedings of the National Academy of Sciences, 85(8), 2444-2448.
24. Touvron H, Bojanowski P, Caron M, et al. ResMLP: Feedforward Networks for Image Classification With Data-Efficient Training [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2023, 45(4): 5314-5321.
25. Kim JH, Jun J, Zhang BT. Bilinear Attention Networks [C]. Advances in Neural Information Processing Systems: Vol. 31. Curran Associates, Inc., 2018.
26. The UniProt Consortium, Bateman A, Martin MJ, et al. UniProt: the universal protein knowledgebase in 2021 [J]. Nucleic Acids Research, 2021, 49(D1): D480-D489.
27. Che C, Zhu M, Zhu Y, et al. A Protein Embedding Model for Drug Molecular Screening [C]. 2020 IEEE International Conference on Big Data and Smart Computing (BigComp). 2020: 251-254.
28. DeepChem [EB/OL]. [2024-05-05]. https://deepchem.io/about/.
29. Python Release Python 3.8.0 [EB/OL]. [2024-05-05].
30. PyTorch [EB/OL]. [2024-05-05]. https://pytorch.org/.
31. RDKit [EB/OL]. [2024-05-05]. https://www.rdkit.org/.
32. NumPy - [EB/OL]. [2024-05-05]. https://numpy.org/.
33. scikit-learn: machine learning in Python — scikit-learn 1.1.3 documentation [EB/OL]. [2022-11-11]. https://scikit-learn.org/stable/index.html.
34. pandas - Python Data Analysis Library [EB/OL]. [2024-05-05]. https://pandas.pydata.org/.
35. torch-geometric · PyPI [EB/OL]. [2024-05-05]. https://pypi.org/project/torch-geometric/.