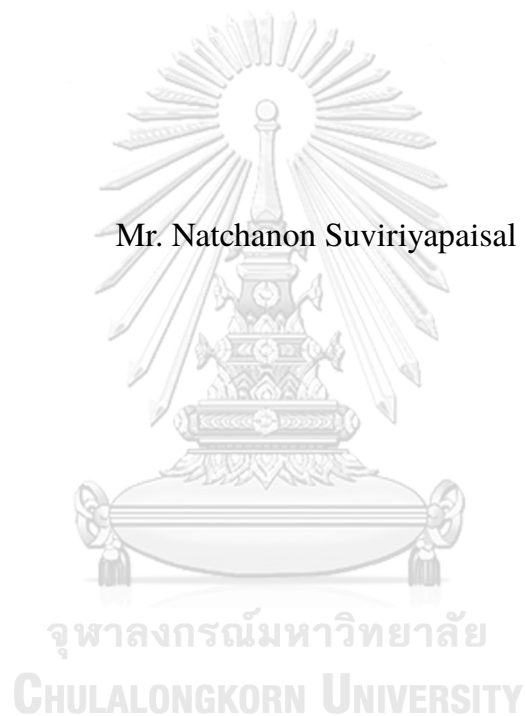


INTEGRATED EDGE FEATURE INTO GRAPH CONVOLUTION
NEURAL NETWORK FOR DRUG-PROTEIN BINDING AFFINITY
PREDICTION

Mr. Natchanon Suviriyapaisal



A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Computer Science

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2023

Copyright of Chulalongkorn University

การรวบรวมคุณลักษณะเส้นเชื่อมระหว่างโหนดในโครงข่ายกราฟคอนโวลูชันเพื่อการ
ทำนายความสามารถในการจับของยาและโปรตีน



วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2566
ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย


Thesis Title INTEGRATED EDGE FEATURE INTO GRAPH CON-
 VOLUTION NEURAL NETWORK FOR DRUG-PROTEIN
 BINDING AFFINITY PREDICTION

By Mr. Natchanon Suviriyapaisal

Field of Study Computer Science

Thesis Advisor Associate Professor Duangdao Wichadakul, Ph.D.

Accepted by the Faculty of Engineering, Chulalongkorn University in Partial
Fulfillment of the Requirements for the Master's Degree



..... Dean of the Faculty of
..... Engineering
(Professor Supot Teachavorasinskun, D.Eng.)

THESIS COMMITTEE

..... Chairman
(Associate Professor Peerapon Vateekul, Ph.D.)

..... Thesis Advisor
(Associate Professor Duangdao Wichadakul, Ph.D.)

..... Examiner
(Dr. Pittipol Kantavat, Ph.D.)

..... External Examiner
(Dr. Thanapat Kangkachit, Ph.D.)

CONTENTS

	Page
Contents	iv
List of Tables	vi
List of Figures	vii
1 Introduction	1
1.1 Motivation	1
1.2 Objectives	2
1.3 Research Plan	2
2 Background	4
2.1 Binding Affinity Measuring Score	4
2.2 Simplified Molecular Input Line Entry System (SMILES)	5
2.3 FASTA Format	6
2.4 Graph Neural Network (GNNs)	7
2.5 Transformer Protein Language Model	9
2.6 Batch Normalization	10
3 Literature review	12
4 Proposed Method	15
5 Experiment setup	23
5.1 Benchmark Dataset	23
5.2 Loss Function And Hyperparameter	24
5.3 Evaluation metrics	25
6 Result	27
References	38

Biography 42



จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

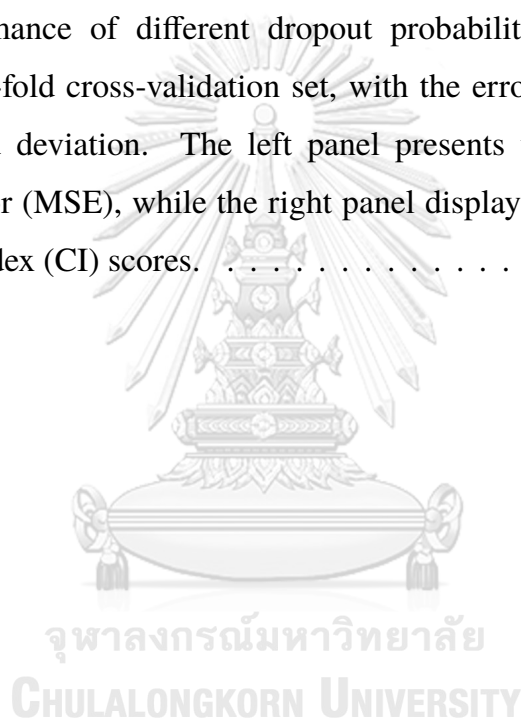
LIST OF TABLES

Table	Page
1.1 Research plan.	3
4.1 Node and edge features for drug compound representation.	17
4.2 Hand-crafted protein features of DGraphDTA (previous work).	18
5.1 Hyperparameter setting.	25
6.1 Model performance on independence test set based on the Davis dataset.	28
6.2 Model performance on independence test set based on the KIBA dataset.	28
6.3 Performance of model component on average five-fold Davis's cross validation set.	32
6.4 Comparison between the edge integration approach using a five-fold Davis's cross validation set.	33
6.5 Comparison between the original DGraphDTA and DGraphDTA with edge integration approach using a five-fold Davis's cross validation set.	33
6.6 Contribution of node-Level and sequence-Level features in combination with a CNN baseline model on a five-fold cross-validation set from Davis's dataset.	34
6.7 Comparison of 1D-GCN performance with predicted contact maps from different contact map prediction models on the average five-fold cross-validation set of Davis's dataset.	35

LIST OF FIGURES

Figure	Page
2.1 An Illustration of how a SMILES sequence can be used to generate a molecule structure (Wikipedia, 2023).	5
2.2 Example of protein sequences in FASTA format (gensas.org, 2023).	6
2.3 (Left) Convolution filter in CNN applied in image data (Right) Convolution filter in GCN applied in graph data (Wu et al., 2021).	8
2.4 The overview of pre-trained feature extraction from protein language model (Elnaggar et al., 2021).	10
4.1 Proposed drug feature extraction and molecular graph construction.	15
4.2 Proposed protein feature extraction and 1D-GCN graph construction.	16
4.3 The overview of proposed model architecture.	16
4.4 Dataset distribution: (Left) Davis dataset contains around 25% (109/442) of the protein sequences longer than 1024 amino acids, (Right) KIBA dataset contains around 20% (42/229) of the protein sequences longer than 1024 amino acids.	19
4.5 Long-sequence protein feature approximation.	20
4.6 Illustration of the propagation rule applied to node v_1 within an existing GCN layer. Here, h represents the hidden state of each node, and W is a learnable weight responsible for transforming the aggregated output into a new hidden state.	20
4.7 Illustration of the propagation rule applied to node v_1 within our Edge-GCN layer. Here, h represents the hidden state of each node, e is an edge feature for each edge, and W is a learnable weight responsible for transforming the aggregated output into a new hidden state.	22
5.1 Model selection procedure using five-fold cross-validation, where the validation result is taken from an average of all folds.	24
5.2 Final evaluation procedure, where the candidate model is chosen from the model with the best performance in cross-validation set.	24

6.1	The training performance of the model varies across different batch normalization configurations, with the model without batch normalization (depicted in blue) exhibiting the most favorable performance on the training set (lower is better).	30
6.2	The validation performance of the model varies across different batch normalization configurations, the model with both BatchNorm and GraphNorm (green) exhibiting the most favorable performance on the validation set (lower is better).	30
6.3	The performance of different dropout probabilities is assessed on Davis's five-fold cross-validation set, with the error bars representing the standard deviation. The left panel presents the averaged mean squared error (MSE), while the right panel displays the averaged concordance index (CI) scores.	31



Chapter I

INTRODUCTION

1.1 Motivation

It takes a long time and expensive cost to ensure the medicine is safe during the conventional drug development procedure. As a past few years, the intensity of COVID-19 has had both direct and indirect impacts on every human worldwide. To stop the disease more quickly, drug repositioning or using drugs for purposes other than their original intent once they have been evaluated can be a big help (Ng et al., 2021; Khataniar et al., 2022). In addition, as machine learning has progressed, numerous computational techniques have been proven helpful in many applications.

Predicting drug-target binding affinity (DTA) is the task that produces a score representing the strength of drug-target pair interaction and can assess how effectively a potential drug binds with a target protein (Jarada et al., 2020). Binding affinity has become a criterion for selecting candidate compounds, which can speed up the drug development process. Using DTA prediction to pick and narrow down the number of compounds for testing in the laboratory can help significantly reduce time and budget. However, the properties of drug and protein in early computational models were manually extracted and required much biological knowledge. Yet, some useful properties, such as edge features, are disregarded in the process. Moreover, because the length of drugs and proteins can be arbitrarily long, the earlier machine learning technique still has a limitation.

To overcome the limitation, this study aims to explore the use of a Graph Convolution Network (GCN) (Kipf and Welling, 2017) for predicting binding affinity score by integrating an edge feature into the computational process and employing the graph propagation technique that can overcome the fixed length problem and accept arbitrary lengths of input in the model training process.

Overall, we expect that the use of GCN for drug target binding affinity prediction will revolutionize the drug discovery process. By improving the accuracy and efficiency of this critical step, researchers can speed up the identification of promising drug candidates, ultimately leading to better treatment options for patients.

1.2 Objectives

The objectives of this study are as follows.

1. To design a deep learning model for binding affinity prediction that is fast and simple, avoid using complex data such as protein structure to train the model to make it simple to use for large-scale virtual screening.
2. To develop a deep learning model that can accurately predict the binding affinity score and rank of drug-protein pairs, which has a performance comparable with the state-of-the-art method.

1.3 Research Plan

1. Explore tools and frameworks (Deep learning).
2. Review the related works.
3. Reproduce the related work results.
4. Explore datasets and feature extraction methods.
5. Experiment new models, evaluate the results, and improve the models' performance.
6. Evaluate the effect of each proposed module.
7. Summarize and publish the research results.
8. Defend the thesis.

Table 1.1: Research plan.

Research Plan	2021			2022												2023					
	10	11	12	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3	4	5	
1. Explore tools and frameworks (Deep learning)																					
2. Review the related works																					
3. Reproduce the related work results																					
4. Explore datasets and feature extraction methods																					
5. Experiment new models, evaluate the results, and improve the model's performance																					
6. Evaluate the effect of each proposed module																					
7. Summarize and publish the research results																					
8. Defend the thesis																					



Chapter II

BACKGROUND

2.1 Binding Affinity Measuring Score

The Inhibition Constant (K_i)

The inhibition constant is a measure of an inhibitor's effectiveness at inhibiting the activity of an enzyme or receptor. It is a constant representing the concentration of the free inhibitor in an equilibrium of 50% inhibition. A lower K_i value indicates that the drug has a stronger binding affinity for the target protein and is more potent at inhibiting its activity.

The Dissociation Constant (K_d)

The dissociation constant measures the dissociation constant at equilibrium as the proportion between k_{off}/k_{on} , where k_{off} is the rate at which the molecule gets out of the protein, and k_{on} is the rate at which a molecule binds to the target protein. The concentration of a drug affects the K_d . The lower the drug concentration, the lower K_d suggests a greater binding strength between the drug and protein.

Half Maximal Inhibitory Concentration (IC_{50})

half maximal inhibitory concentration (IC_{50}) measures the drug's total concentration required to inhibit 50% of the target protein's activity. It is often used to measure a drug's potency and efficacy in inhibiting the protein's activity. A lower IC_{50} value indicates the drug is more potent and effective at inhibiting the target protein's activity.

In summary, K_i , K_d , and IC_{50} are all essential measures of drug-protein interactions that provide information on the strength of the binding affinity between a

drug and its target protein and the potency and efficacy of the drug in inhibiting the activity of the protein (scienesnail.com, 2019).

2.2 Simplified Molecular Input Line Entry System (SMILES)

The simplified molecular-input line-entry system, also known as SMILES (Weininger, 1988), is a string representation of a chemical compound's molecular structure. The SMILES format encodes and shares simple and clear structural information about chemical molecules.

In the SMILES system, atomic symbols represent atoms, whereas symbols such as "-", "=", and "#" represent bonds to signify single, double, and triple bonds, respectively. Based on the interconnectedness of the atoms in the molecule, the atoms and bonds are then arranged as a linear string. Figure 2.1 shows an example of how to construct a SMILES string.

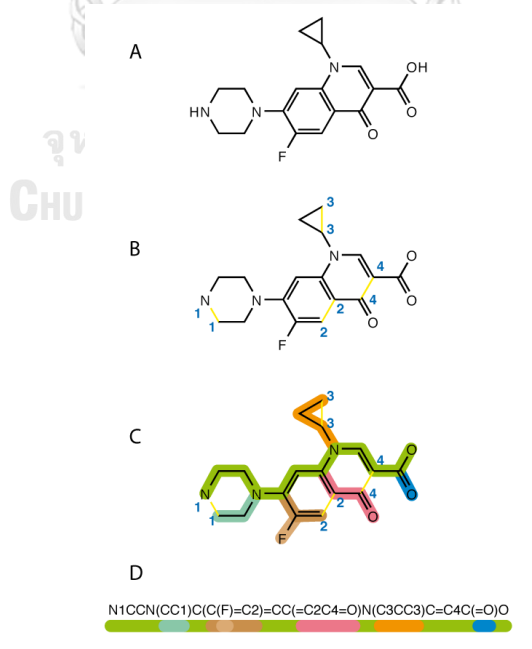
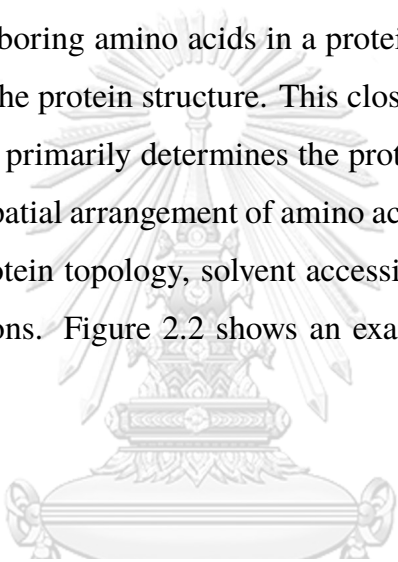


Figure 2.1: An Illustration of how a SMILES sequence can be used to generate a molecule structure (Wikipedia, 2023).

2.3 FASTA Format

The FASTA format begins with a single-line description signaled by a greater-than symbol (“>”), followed by a brief sequence description. The protein sequence, expressed as a string of letters, follows the description line. The letters stand for the single-letter amino acid codes of the protein sequence, such as A for alanine, C for cysteine, and D for aspartic acids. There are 20 typical amino acids used frequently in protein sequences.

In general, neighboring amino acids in a protein sequence are more likely to be physically close in the protein structure. This closeness is because the linear sequence of amino acids primarily determines the protein’s folding. However, many factors can affect the spatial arrangement of amino acids in a protein structure, such as local and global protein topology, solvent accessibility, hydrogen bonding, and non-covalent interactions. Figure 2.2 shows an example of a FASTA protein sequence format.



```
>Sequence_1 assembly1
CCCTAAACCCCTAAACCCCTAAACCCCTGAATCCTTAATCCCTAAATCCCTAAAT
CTTTAAATCCTACATCCATGAATCCCTAAATACCTAATCCCTAAACCCGAAACCGGTTT
CTCTGGTTGAAAATCATTGTGTATATAATGATAATTTATCGTTTTTATGTAATTGCTTA
TTGTTGTGTGTAGATTTTTAAAAATATCATTGAGGTCATAAATCCTATTTCTTGT
GGTTTTCTTTCCCTTCACTTAGCTATGGATGGTTTATCTTCATTTGTTATATTGGATACAA
GCTTTGCTACGATCTACATTTGGGAATGTGAGTCTCTTATTGTAACCTTAGGGTTGGTTT
ATCTCAAGAATCTTATTAATTGTTTGGACTGTTTATGTTTGGACATTTATTGTCATTCTT
>Sequence_2
CCCTAAACCCCTAAACCCCTAAACCCCTGAATCCTTAATCCCTAAATCCCTAAAT
CTTTAAATCCTACATCCATGAATCCCTAAATACCTAATCCCTAAACCCGAAACCGGTTT
CTCTGGTTGAAAATCATTGTGTATATAATGATAATTTATCGTTTTTATGTAATTGCTTA
TTGTTGTGTGTAGATTTTTAAAAATATCATTGAGGTCATAAATCCTATTTCTTGT
GGTTTTCTTTCCCTTCACTTAGCTATGGATGGTTTATCTTCATTTGTTATATTGGATACAA
GCTTTGCTACGATCTACATTTGGGAATGTGAGTCTCTTATTGTAACCTTAGGGTTGGTTT
ATCTCAAGAATCTTATTAATTGTTTGGACTGTTTATGTTTGGACATTTATTGTCATTCTT
```

Figure 2.2: Example of protein sequences in FASTA format (gensas.org, 2023).

2.4 Graph Neural Network (GNNs)

Message Passing In GNNs

A Graph Neural Network is designed to function on graph-structured data. Nodes (vertices) and edges connecting nodes make up a graph. GNNs use these nodes and edges to discover patterns and relationships in graph data.

GNNs are useful for numerous applications, such as node classification, link prediction, graph categorization, and recommendation systems. In addition to social network analysis, drug discovery, and computer vision, they have been effectively implemented in various fields.

GNNs use a message-passing rule to propagate the information in the learning process. Given a graph $G = (V, E, X)$, where V is a set of vertices, E is a set of edges and X are the node features. The message-passing rule is a function that takes as input the features of a node and its neighbors and produces a message that is passed from the neighbors to the node. This message is then aggregated with the node's feature representation to produce a new feature vector for the node, as shown in Equation 2.1 with a simple summation as an aggregation function (Xu et al., 2018).

$$m_v^{(t)} = \sum_{u \in N(v)} h_u^{(t)} \quad (2.1)$$

Suppose $h_u^{(t)}$ represents node embeddings for some vertex u at iteration t . $m_v^{(t)}$ is an output message of vertex v , and $N(v)$ is a set of neighboring nodes of node v . Then, the output message is passed to the update function that updates a node's feature vector based on the aggregated messages from its neighbors. The update rule combines the original feature vector of the node with the aggregated messages to generate a new feature vector for the node $h_v^{(t+1)}$ depending on the design of the GNN, but some common methods include concatenation, addition, or more complex operations such as learnable weight. The message update with arbitrary update

function \emptyset is shown in Equation 2.2.

$$h_v^{(t+1)} = \emptyset \left(h_v^{(t)}, m_v^{(t)} \right) \quad (2.2)$$

Graph Convolutional Network (GCN)

Similar to how convolutional neural networks (CNNs) learn feature representations of local patches in images (Figure 2.3), GCNs employ convolutional filters learnable weight to learn localized feature representations of each node in a graph. The filters are applied to each node's neighborhood, which consists of the node itself and its adjacent neighbors. The propagation rule of GCN is as follows: message passing, aggregation, and feature update, which can be formulated as Equation 2.3 (Kipf and Welling, 2017).

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)}) \quad (2.3)$$

In this context, \tilde{A} denotes the adjacency matrix of a graph featuring a self-loop connection. \tilde{D} corresponds to the diagonal node degree matrix of \tilde{A} , serving to measure the degree of each node and preserves the scale of the output feature vector. $H^{(l)}$ signifies the output from a previous hidden layer. $W^{(l)}$ stands for the learnable parameters, and σ represents an activation function, such as ReLU.

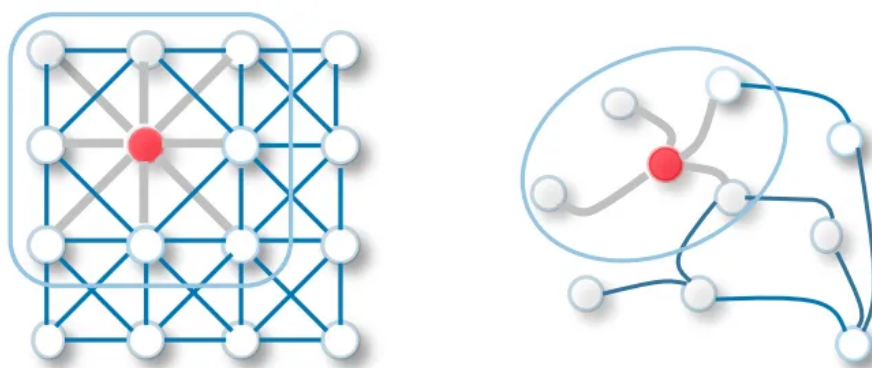


Figure 2.3: (Left) Convolution filter in CNN applied in image data (Right) Convolution filter in GCN applied in graph data (Wu et al., 2021).

2.5 Transformer Protein Language Model

The transformer protein language model is a variant of the transformer architecture (Vaswani et al., 2017) designed explicitly for protein sequences. It is a deep learning model that uses self-attention mechanisms to capture the relationships between amino acids in a protein sequence, enabling it to generate or accurately predict a sequence of amino acids.

Evolutionary Scale Modeling (ESM) (Meier et al., 2021) and ProtTrans (Elnaggar et al., 2021) are examples of state-of-the-art models that worked on a transformer protein language model utilizing the capability of self-supervised learning that can learn the representation on an unlabeled dataset, allowing this technique to apply to a massive unlabeled amino acid sequence dataset from a variety of sources, such as the Uniparc database (UniProt Consortium, 2007) and the BFD dataset (Steinegger et al., 2019). Like a natural language model, the transformer may learn a sequence of amino acid elements as a token where self-supervised is achieved by the use of a masked language model that corrupts tokens by substituting them with special tokens, and then the network is trained to predict the missing tokens from the corrupted sequences.

The pre-trained feature is extracted through a model embedding vector, such as the intermediate hidden layer of the protein language model (Figure 2.4), and then this representation vector is used as an input for other tasks. In addition, they assess the pre-trained model's performance by predicting the protein's secondary and tertiary structures. Consequently, the prediction result is significantly more accurate when pre-trained features are utilized, suggesting that pre-trained features may contain information about protein structure.

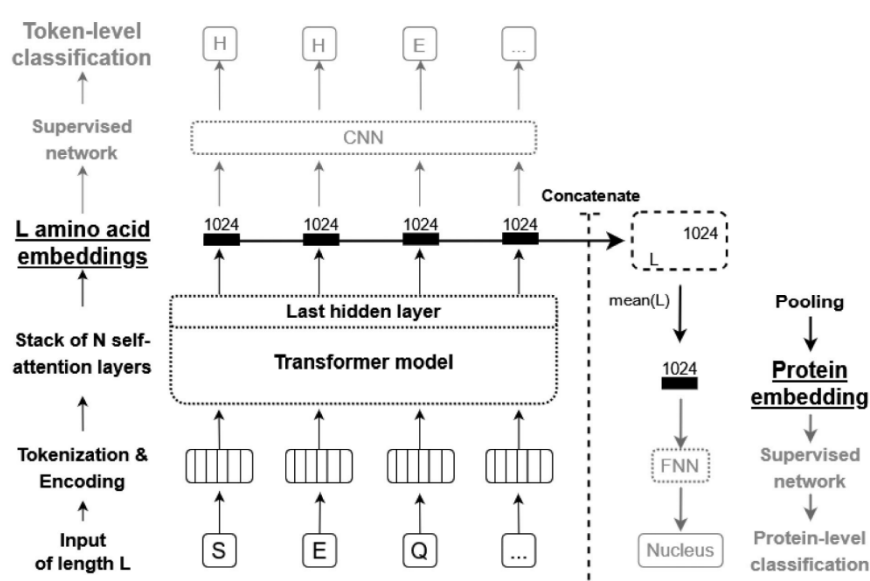


Figure 2.4: The overview of pre-trained feature extraction from protein language model (Elnaggar et al., 2021).

2.6 Batch Normalization

Due to memory and computational constraints, it may be impractical to train a model and compute the gradient over the entire dataset; therefore, smaller mini-batches (a small subset of the training data) may be used in practice. However, there is a disadvantage to using a mini-batch because a model learns and updates its weight based on a small sample of the entire dataset, and the distribution of the sample dataset is continuously changed for each mini-batch. This change can result in the learning algorithm attempting to pursue a moving target. The changes in distribution are referred to as "internal covariate shifts."

Batch normalization is a technique used in deep learning to enhance the training of neural networks by normalizing each layer's inputs. During the training of a neural network, the distribution of inputs to each layer changes, which can slow down the training process. Batch normalization addresses this issue by standardizing the inputs to each layer, thereby stabilizing and accelerating the training process.

Batch normalization (Ioffe and Szegedy, 2015) involves calculating the mean and variance of the inputs in a mini-batch for each layer and using these values to normalize the inputs by making the mini-batch distribution have a mean of zero and a variance of one. Batch normalization specifically applies the following Equation 2.4 to a mini-batch of inputs:

$$z = \frac{x - \mathbf{E}[x]}{\sqrt{\mathbf{Var}[x] + \epsilon}} * \gamma + \beta \quad (2.4)$$

where x is the input to a layer, the mean (Expectation \mathbf{E}) and variance (\mathbf{Var}) are calculated per-dimension over the mini-batches, β and γ are learnable parameters, ϵ is a small float added to avoid dividing by zero, and z is the normalized output

In addition to improving the training of neural networks, batch normalization has been shown to have other benefits, such as reducing overfitting and improving generalization performance, as it tries to estimate expectations and variance from many small subsets of data.

Overall, batch normalization has several benefits, including faster training times, improved generalization performance, and increased robustness to changes in the input distribution. It also reduces the need for dropout and weight decay techniques, commonly used to prevent overfitting in deep learning models.

Chapter III

LITERATURE REVIEW

This chapter will summarize the current research results that use deep learning techniques for drug target binding affinity prediction.

SimBoost

SimBoost (He et al., 2017) is a machine-learning-based method that uses similarity-based features. SimBoost first calculates a similarity matrix that measures the similarity between each pair of drugs and targets using matrix factorization (MF), which is very effective on recommendation tasks. The similarity matrix is then used to generate the feature vector that feeds the gradient-boosting machine model and the affinity score it has learned to predict.

DeepDTA

DeepDTA (Öztürk et al., 2018) is the prior work employing a Convolutional neural network (CNN) to process the drug and target protein sequences and then merge the feature vectors to predict their binding affinity. It has shown superior performance in predicting binding affinity compared to traditional machine learning methods such as SimBoost.

MT-DTI

MT-DTI (Shin et al., 2019) utilized a natural language processing (NLP) technique called "Transformer" Vaswani et al. (2017) for the drug molecule to take advantage of the self-attention mechanism that can be used to capture relationships across the sequences.

GraphDTA

GraphDTA (Nguyen et al., 2020) employed a graph-based technique to perform the task of affinity prediction. By converting the representation of drug compounds from a text sequence to a graph structure, they investigated many variants of graph neural networks (GNNs) to find the best GNN variant for this task. However, for the protein branch, which informative feature is far more complex than the drug's compound, the same CNN technique as DeepDTA was used.

DgraphDTA

DGraphDTA (Jiang et al., 2020) model expanded upon the GraphDTA by representing both drug compounds and protein sequences as a 2D graph by including a protein structure prediction method called contact-map prediction. Then, drug and protein graphs are fed to the independent GCN layer to extract their representation.

GEFA

GEFA (Nguyen et al., 2022) employed an "early fusion" strategy, which means that the model combined drug and target node input features first via a self-attention mechanism to reflect a protein's change in representation during the binding process, and then a new representation was used as an input for GCN.

WGNN

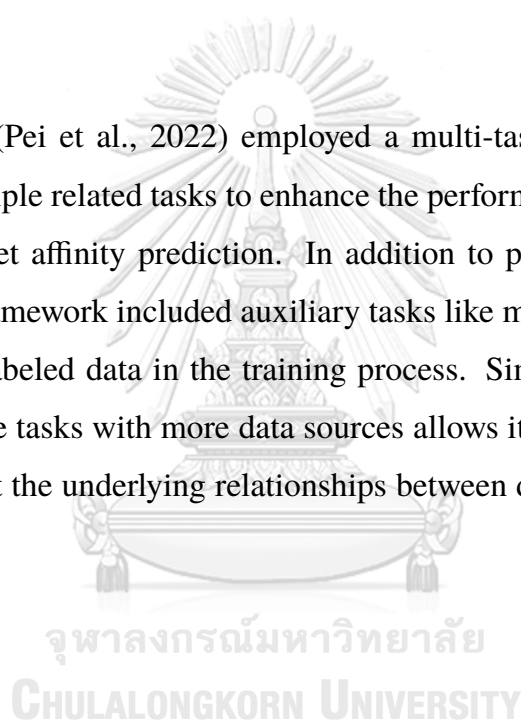
WGNN (Jiang et al., 2022) focused on improving the speed and accuracy of their previous work, DgraphDTA, by avoiding the use of complex and time-consuming features such as PSSM (position-specific scoring matrix), which is extracted from MSA (multiple sequence alignment), and by providing a more informative structure of proteins by using the confidence score of contact map prediction as a weighted score of graph edges (WGNN), considered as the importance of amino acid connection.

MGraphDTA

MGraphDTA (Yang et al., 2022) proposed a deep multiscale graph neural network architecture consisting of a multiscale graph neural network (MGNN) in the compound branch and a multiscale convolution neural network (MCNN) in the protein branch, each of which captures information at various scales of granularity. These layers allow the model to learn hierarchical representations of the drug-target interaction that incorporate local and global information.

SMT-DTA

SMT-DTA (Pei et al., 2022) employed a multi-task learning framework to jointly learn multiple related tasks to enhance the performance of the primary task, namely drug-target affinity prediction. In addition to predicting affinity with labeled data, the framework included auxiliary tasks like masked language modeling (MLM) with unlabeled data in the training process. Simultaneously, training the model on multiple tasks with more data sources allows it to learn shared representations that reflect the underlying relationships between drugs, protein targets, and properties.



Chapter IV

PROPOSED METHOD

This chapter will explain the components and methodology of the proposed deep learning framework architecture.

In our study, the proposed model takes two inputs: a drug compound and a protein sequence represented in SMILES format and FASTA format, which are then transformed into graph structures. The drug branch (Edge-GCN layer) constructs a graph using the actual contacts, with edges representing atom-to-atom bonds (Figure 4.1).

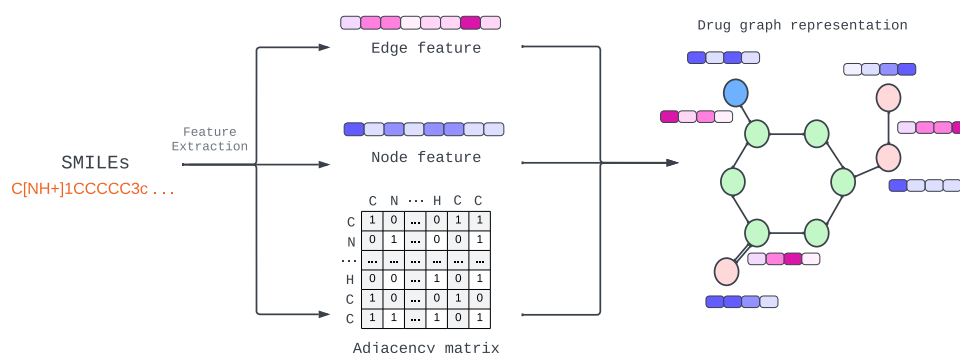


Figure 4.1: Proposed drug feature extraction and molecular graph construction.

In contrast, the protein branch (GCN layer) generates a 1D graph from the sequence by connecting each amino acid to its neighboring amino acids, thereby avoiding complex contact information that would require more computational resources. Since the length of input for drugs and proteins can vary, we utilized the whole sequence for feature extraction and model computation using the GCN approach, which can take an arbitrary-length input rather than a fixed-length technique like padding or truncating, as required by CNN and NLP. Additionally, to handle potentially lengthy protein sequences, requiring many GCN layers for comprehensive information gathering (since a single GCN can obtain the information with a

single hop), we added an additional branch in the form of a Linear layer. This branch is responsible for extracting a comprehensive, sequence-level feature by taking the average of node-level features from sub-sequences within the entire sequence and subsequently passing them through a linear layer for representation learning (Figure 4.2).

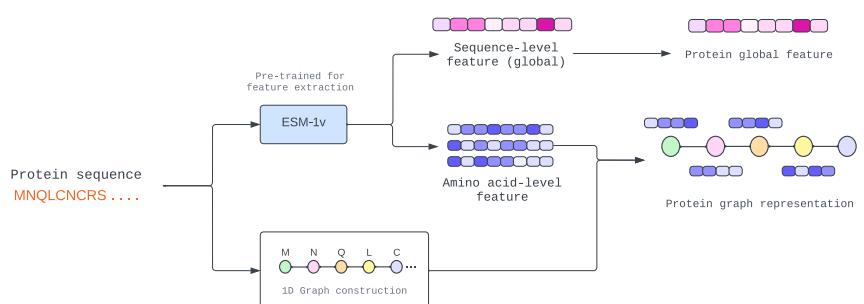


Figure 4.2: Proposed protein feature extraction and 1D-GCN graph construction.

We predicted the binding affinity score, a continuous value, by first employing GCN as a feature extraction layer to extract a latent vector that learns node-level information and then passing this latent vector to a fully connected layer for prediction (Figure 4.3).

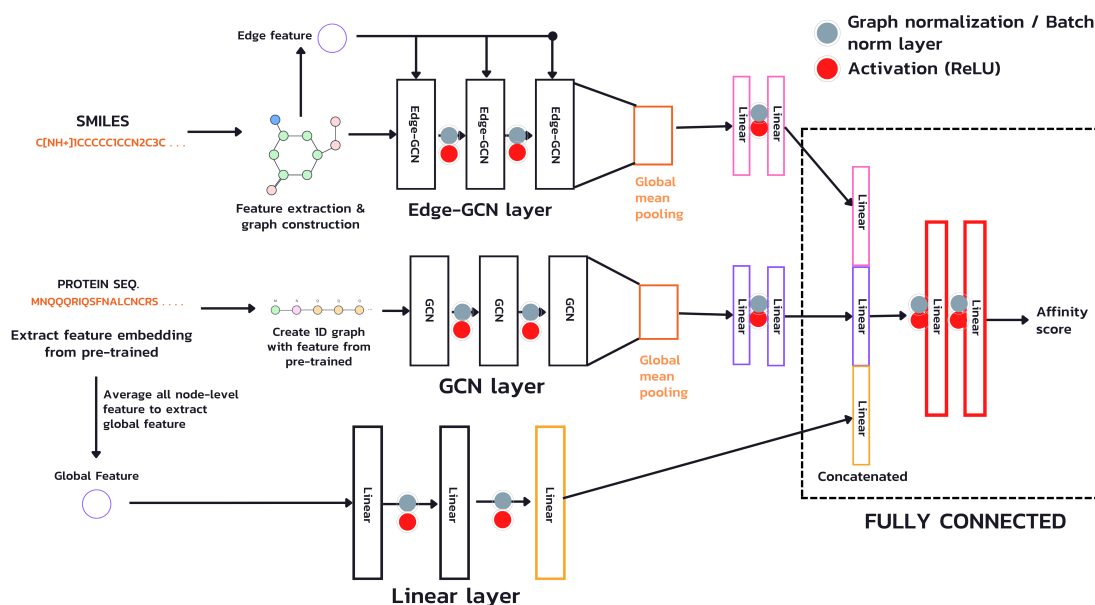


Figure 4.3: The overview of proposed model architecture.

Drug representation

Our model expects drug compounds in SMILE format, and we use the TorchDrug (Zhu et al., 2022) and RDKit libraries (Landrum et al., 2023) to extract the node and edge features, including an adjacency matrix to construct a graph. Atomic number, formal charge, explicit hydrogen, chiral tag, and radical electron are part of the one-hot vector representing a node's attributes. Compared to previous research, we extract edge characteristics, which significantly increase the complexity and insight of our data. Instead of using edges to link nodes or weighted edges to convey the importance of each edge, as was done in some previous work, we integrated a multidimensional edge feature into each convolution layer where our edge feature comprises bond type, bond stereo, and stereo atom. The lists of node and edge features used in this work are shown in Table 4.1.

Table 4.1: Node and edge features for drug compound representation.

Name	Definition	Size
Node feature (66)		
Atom symbol (char)	Atomic symbol	18
Atomic chiral tag (type)	Type of chirality	4
Degree of atom (integer)	Number of directly-bonded neighbors in the molecule including Hs	8
Number of formal charges (integer)	Number of formal charges in the molecule	11
Number of explicit and implicit Hs (integer)	Total number of Hs (implicit and explicit) that this atom is bound to	7
Number of radical electron (integer)	Number of unpaired electrons for an atom (radical)	8
Atom hybridization (type)	Orbital hybridisation, introduced to explain molecular structure	8
Is aromatic (bool)	Whether the atom is aromatic (molecule has cyclic; ring-shaped structures)	1
Is in ring (bool)	Whether the atom is in an aromatic ring	1
Edge feature (18)		
Bond type (type)	A type of bond between atoms	4
Bond direction (type)	The direction of the bond	7
Bond stereo configuration (type)	The stereo configuration of the bond	6
Is bond conjugated (bool)	Whether the bond is considered to be conjugated	1

Protein representation

The most challenging protein-related problem is how to derive the essential protein characteristics from the string sequence alone. This challenge resulted from proteins being much more complicated than drugs, of which the structures are described in SMILE format. The properties of individual amino acids, such as residue-symbol, aliphatic, and polarity, allow us to infer some aspects of a protein. However, when these amino acids form a protein, a complex structure is generated, and the function can be altered. In previous work, DGraphDTA contains 14 hand-crafted features (Table 4.2) representing each amino acid node feature. These features, however, may not be sufficient to express a protein's function and structure. In this work, we are investigating and evaluating the effect of using a pre-trained model called

”Evolutionary Scale Modeling,” or ”ESM” (Meier et al., 2021), a transformer protein language model, to extract the most valuable and dependable protein feature from its sequence.

Table 4.2: Hand-crafted protein features of DGraphDTA (previous work).

Number	Feature	Dimension
1	One-hot encoding of the residue symbol	21
2	Position-specific scoring matrix (PSSM)	21
3	Whether the residue is aliphatic	1
4	Whether the residue is aromatic	1
5	Whether the residue is polar neutral	1
6	Whether the residue is acidic charged	1
7	Whether the residue is basic charged	1
8	Residue weight	1
9	The negative logarithm of the dissociation constant for the $-COOH$ group	1
10	The negative logarithm of the dissociation constant for the $-NH_3$ group	1
11	The negative logarithm of the dissociation constant for any other group in the molecule	1
12	pH at the isoelectric point	1
13	Hydrophobicity of residue (pH = 2)	1
14	Hydrophobicity of residue (pH = 7)	1

However, there is a limitation to using pre-trained models, which require a protein sequence length of fewer than 1024 amino acids. A large portion of our protein sequence data is longer than 1000 amino acids (Figure 4.4). Therefore, we need a solution to approximate long-sequence protein characteristics. One method is partitioning a protein sequence into sub-sequences, which overlap the others for a better approximation, and the final sequence feature is an average of all sub-sequences (described in Figure 4.5 and Algorithm 1).

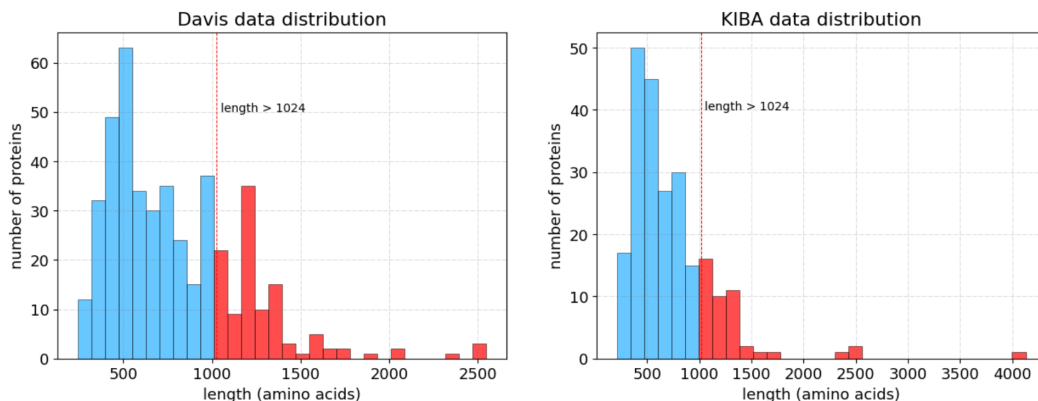


Figure 4.4: Dataset distribution: (Left) Davis dataset contains around 25% (109/442) of the protein sequences longer than 1024 amino acids, (Right) KIBA dataset contains around 20% (42/229) of the protein sequences longer than 1024 amino acids.

Moreover, as mentioned in the model overview earlier, given the complex nature of proteins, it presents a significant challenge to deduce their overall representation solely from the node-level data. To address this, we utilized a pre-trained model trained on a massive dataset with a highly complex model. Our approach involved extracting a global feature by computing the average node feature from this pre-trained model. This average node feature resulted in a per-sequence feature of 1280 dimensions. Subsequently, we passed this feature through three fully connected layers before concatenating it with the structural representation of the drug and protein sequences.

Integrated multi-dimensional edge feature

Through the enhancement of the graph message passing procedure, we were able to improve the representation of a drug compound that had additional multi-dimensional edge features, which were extracted during the feature extraction step. In order to achieve this improvement, the GCN propagation rule will be modified to include edge features at every convolution step, as shown in Equation 4.1, and is referred to as Edge-GCN.

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}(H^{(l)} + EW_e^{(l)})W^{(l)}) \quad (4.1)$$

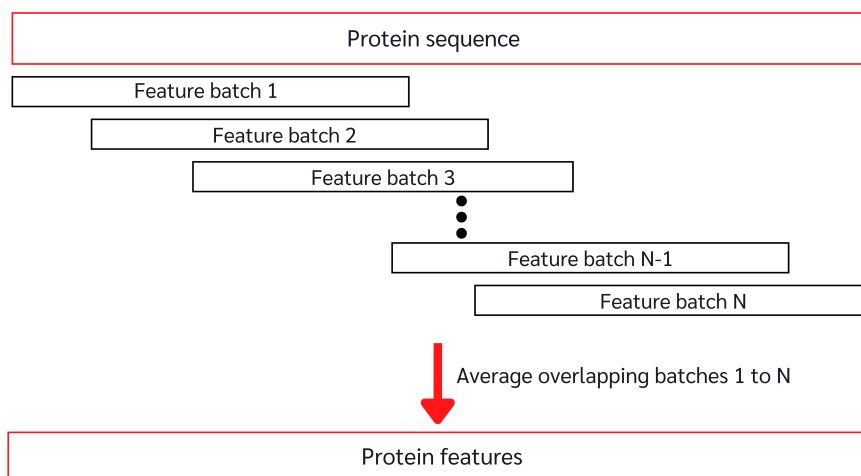


Figure 4.5: Long-sequence protein feature approximation.

In Equation 4.1, $H^{(l)}$ in Equation 2.3 is replaced with $H^{(l)} + EW_e^{(l)}$, where E is a multi-dimensional edge feature and $W_e^{(l)}$ is a learnable parameter which serves the purpose of transforming the edge feature dimension into the node feature dimension.

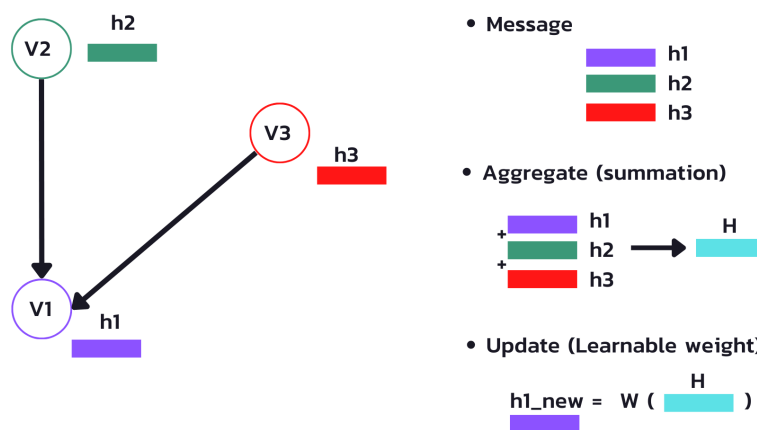


Figure 4.6: Illustration of the propagation rule applied to node v_1 within an existing GCN layer. Here, h represents the hidden state of each node, and W is a learnable weight responsible for transforming the aggregated output into a new hidden state.

Figures 4.6 and 4.7 depict a comparison between our Edge-GCN and an existing GCN where Edge-GCN integrated the hidden state of the neighboring node with an edge feature, transformed it into a node dimension via learnable weight, and

Algorithm 1 Overlapping protein's feature approximation

Input: protein sequence: seq
Initialization: protein feature $\leftarrow zeros(Len(seq), 1280)$
 1: **if** $Len(seq) \leq 1024$ **then**
 2: protein feature $\leftarrow ESM-1v(seq)$
 3: **Return:** protein feature
 4: **else**
 5: $stride \leftarrow 5$
 6: $weights \leftarrow zeros(Len(seq), 1280)$
 7: $windows \leftarrow 500$
 8: $N \leftarrow (Len(seq) - windows) / stride$
 9: **for** $i = 0$ **to** N **step** 1 **do**
 10: $start \leftarrow i * stride$
 11: $end \leftarrow \min(start + windows, Len(seq))$
 12: $subseq \leftarrow seq[start : end]$
 13: $subseq\ feature \leftarrow ESM-1v(subseq)$
 14: $protein\ feature[start : end] \leftarrow protein\ feature[start : end]$
 15: $+ subseq\ feature$
 16: $weights[start : end] \leftarrow weights[start : end] + 1$
 17: **end for**
 18: $protein\ feature \leftarrow protein\ feature / weights$
 19: **Return:** protein feature
 20: **end if**

implemented the same propagation rules as GCN.

In this architecture, both drug and protein sequences are first transformed into graph representations, and then they are fed through two different networks, each of which is composed of three-layer graph convolutional networks for the purpose of extracting representations. Following the implementation of these networks, a global pooling layer is introduced in order to guarantee that the feature dimension is consistent. Various hyperparameters are available for global pooling operations, including options such as sum, mean, or max. In this work, we adopted global mean pooling as recommended in the DGraphDTA experiment, which is defined by the Equation 4.2.

$$r_i = \frac{1}{N_i} \sum_{n=1}^{N_i} x_n \quad (4.2)$$

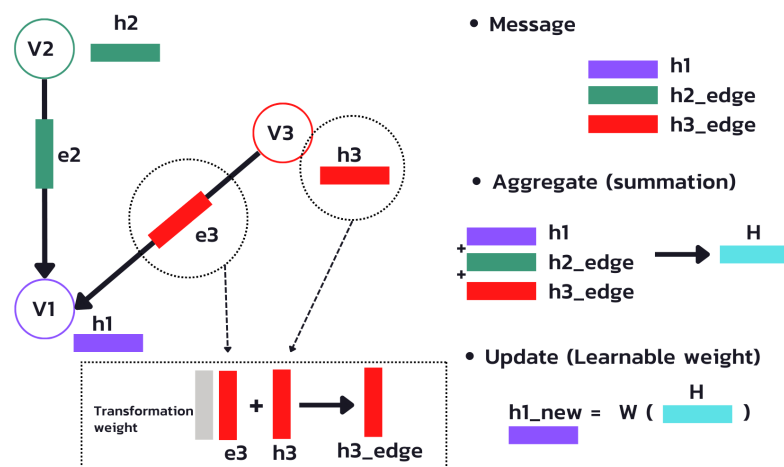


Figure 4.7: Illustration of the propagation rule applied to node v_1 within our Edge-GCN layer. Here, h represents the hidden state of each node, e is an edge feature for each edge, and W is a learnable weight responsible for transforming the aggregated output into a new hidden state.

Where x represents the feature matrix corresponding to each node in the GCN output, and N denotes the number of nodes in the graph. Subsequently, the model acquires a latent representation sized $(1, F)$, with F indicating the number of output channels for the final layer..

In this work, Batch normalization (BatchNorm) (Ioffe and Szegedy, 2015) was incorporated after each output layer in the fully connected layers, while graph normalization (GraphNorm) (Cai et al., 2020) was applied to the GCN layers. We used the Rectified Linear Unit (ReLU) as an activation function. Lastly, we combined the drug's latent features, the protein's latent features, and the protein's global features and passed them through two fully connected layers to make predictions regarding the affinity score.

Chapter V

EXPERIMENT SETUP

5.1 Benchmark Dataset

To evaluate how well our model performs compared to others, we used the Davis dataset (Davis et al., 2011), which includes 30056 drug-target interactions for 68 inhibitors and 442 protein kinases. The interaction intensity is quantified by the dissociation constant (Kd). The Kd value was translated into a log scale called pKd in the same manner used in DeepDTA and other previous works.

Additionally, the KIBA dataset (Tang et al., 2014) includes 246088 drug-target interactions of 52498 unique inhibitors and 467 protein kinases made from different bioactivity sources, such as Ki, Kd, and IC50. He et al. (2017) filtered the original KIBA dataset by removing all medicines and proteins with fewer than ten interactions. We used this filtered KIBA dataset, containing 2111 and 229 drugs and proteins with 118254 drug-target interactions for model training.

An approach known as five-fold cross-validation (illustrated in Figure 5.1) was utilized for the purpose of evaluating and selecting candidate models. The candidate model was selected based on the best-averaged performance across all of the validation sets for each fold. After that, the model that was selected was evaluated on the independence test set, and predictions were made by utilizing the weights that were obtained from each particular fold. The final outcome, which is illustrated in Figure 5.2, was achieved by calculating the average of the test results obtained from each fold.

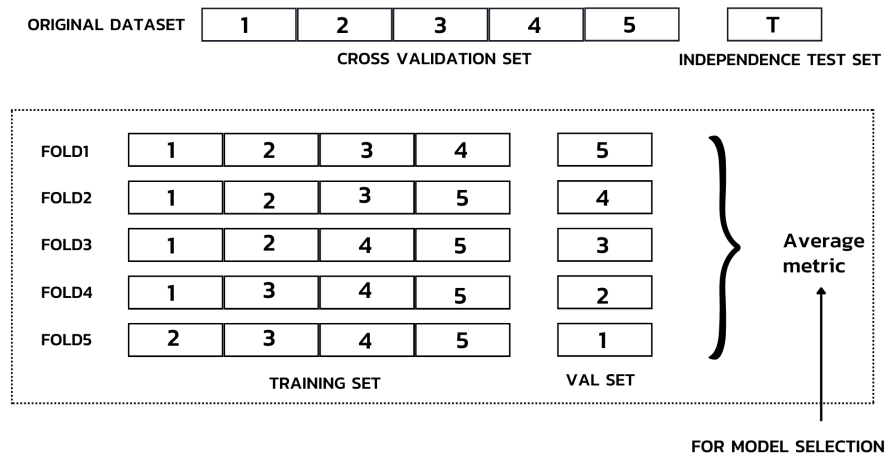


Figure 5.1: Model selection procedure using five-fold cross-validation, where the validation result is taken from an average of all folds.

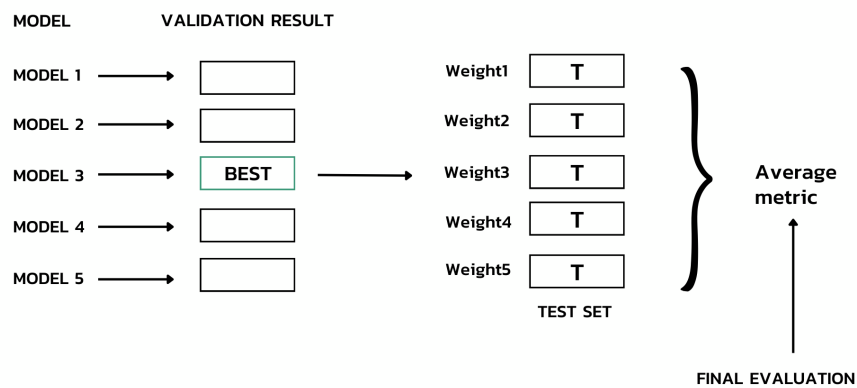


Figure 5.2: Final evaluation procedure, where the candidate model is chosen from the model with the best performance in cross-validation set.

5.2 Loss Function And Hyperparameter

During the training process, our primary objective was to minimize the differences between the predicted affinity score generated by our model and the true affinity score. Given that our objective involves regression, we choose the Mean

Square Error (MSE) as our preferred loss function. The MSE measures the average of the squared differences between the predicted (P) and actual values (Y) as shown in Equation 5.1. providing a metric that aligns with our regression-focused training objective.

$$MSE = \frac{1}{n} \sum_{i=1}^n (P_i - Y_i)^2 \quad (5.1)$$

We employed AdamW optimization (Loshchilov and Hutter, 2017), an optimization approach that enhances regularization by disentangling weight decay from the gradient update. We also adjusted the weight decay parameter of both datasets to 0.01 for overfit prevention. To facilitate the learning during the late iteration process, we also employed the learning rate decay, which begins at 0.001 and decreases to 80% for every 100 epochs. Note that we manually tuned most hyperparameters because the training process takes a significant amount of time to complete. The summary of the hyperparameter setting is shown in Table 5.1.

Table 5.1: Hyperparameter setting.

hyperparameter	setting
epoch	1000
optimizer	AdamW
batchsize	128
learning rate	0.001
weight decay	0.01
latent size after GCN	128

5.3 Evaluation metrics

In order to ensure consistency with the baseline and other state-of-the-art methods, we utilized the same evaluation metrics consisting of the Mean Square Error (MSE) and the Concordance Index (CI), where both were used as the main metrics for DTA tasks. The Mean Squared Error (MSE), frequently employed in regression situations, measures the difference between predicted and actual values. A smaller MSE indicates that our model's predictions are closer to the actual values. Conversely, the CI score prioritizes the ordering of predictions rather than their predicted values, as defined by the equation 5.2.

$$CI = \frac{1}{Z} \sum_{d_i > d_j} h(b_i - b_j) \quad (5.2)$$

$$h(x) = \begin{cases} 1, & \text{If } x > 0 \\ 0.5, & \text{If } x = 0 \\ 0, & \text{If } x < 0 \end{cases} \quad (5.3)$$

The variable b_i denotes the expected value for the higher affinity d_i , whereas b_j reflects the predicted value for the smaller affinity d_j . The symbol Z denotes a normalization constant, while the function $h(x)$ corresponds to the step function, as seen in Equation 5.3.

The Pearson correlation coefficient, as defined in Equation 5.4, assesses the linear association between two variables—in our case, the predicted and ground truth values. Here, cov signifies the covariance between the predicted value (p) and the actual value (y), while $\sigma(p)$ and $\sigma(y)$ represent the standard deviations of the predicted and actual values, respectively.

$$Pearson = \frac{cov(p, y)}{\sigma(p)\sigma(y)} \quad (5.4)$$

Chapter VI

RESULT

This chapter will show the experimental results of each component described in the proposed method chapter, along with a preliminary result compared to other related work.

Performance evaluation

We conducted a comparative analysis between our approach and the methodologies employed in previous research. In order to ensure an appropriate comparison, we analyzed the performance on a separate test set containing data that had not been seen before. We then reported the assessment metrics, which included Mean Squared Error (MSE), Concordance Index (CI), and Pearson correlation coefficient along with its standard deviation. It is important to acknowledge that we replicated the results of some previous research in order to ensure fairness.

In Tables 6.1 and 6.2, We present the results of our study, where we conducted a comparison between our model and various existing approaches, including CNN-based, graph-based, and transformer-based, to estimate drug and protein affinity scores. Our method, incorporating Edge-GCN, 1D-GCN, and global features, demonstrated significant improvements across all metrics (MSE, CI, and Pearson correlation) on the Davis dataset. Specifically, we achieved values of 0.216 for MSE, 0.897 for CI, and 0.855 for Pearson correlation on the independence test set. Our model also exhibited superior performance on two metrics on the KIBA dataset compared to other methods, achieving the lowest MSE and the highest Pearson correlation value. Additionally, it achieved the highest Concordance Index (CI) score among graph-based approaches, on par with MGraphDTA. However, the transformer-based model SMT-DTA surpassed all with the highest CI score of 0.894.

Notably, our model surpassed existing graph-based techniques, including GEFA, which previously achieved the lowest MSE of 0.228 on the Davis dataset. It's crucial to highlight that our model achieves this level of performance without requiring knowledge of protein structure. This sets it apart from other graph-based methods that generate protein graphs based on predicted contact maps. This achievement suggests that our pre-trained model, particularly ESM, may acquire an understanding of protein function or certain structural information during its training process.

Table 6.1: Model performance on independence test set based on the Davis dataset.

Model	Compound	Protein	MSE (std)	CI (std)	Pearson (std)
SimBoost (He et al., 2017)	MF	MF	0.282	0.872(0.002)	-
DeepDTA (Öztürk et al., 2018)	CNN	CNN	0.261	0.878(0.004)	-
MT-DTI (Shin et al., 2019)	Transformer	CNN	0.245	0.887(0.003)	-
GraphDTA (Nguyen et al., 2020)*	GIN	CNN	0.251(0.003)	0.882(0.003)	-
DGraphDTA (Jiang et al., 2020)*	GCN	GCN(contact)	0.238(0.005)	0.888(0.004)	0.840(0.003)
GEFA (Nguyen et al., 2022)	GCN	GCN(contact)	0.228	0.893	0.846
WGNN (Jiang et al., 2022)*	GNN	WGNN(contact)	0.244(0.004)	0.888(0.002)	0.837(0.002)
MGraphDTA (Yang et al., 2022)*	MGNN	MCNN	0.233(0.005)	0.885(0.004)	0.843(0.004)
SMT-DTA (Pei et al., 2022)	Transformer	Transformer	0.219	0.890	-
Our (iEdgeDTA)	Edge-GCN	1D-GCN	0.216(0.004)	0.897(0.001)	0.855(0.002)

* these results were taken from our reproduction.

Table 6.2: Model performance on independence test set based on the KIBA dataset.

Model	Compound	Protein	MSE (std)	CI (std)	Pearson (std)
SimBoost (He et al., 2017)	MF	MF	0.222	0.836(0.001)	-
DeepDTA (Öztürk et al., 2018)	CNN	CNN	0.194	0.863(0.002)	-
MT-DTI (Shin et al., 2019)	Transformer	CNN	0.152	0.882(0.001)	-
GraphDTA (Nguyen et al., 2020)*	GCN	GCN	0.186(0.009)	0.871(0.001)	-
DGraphDTA (Jiang et al., 2020)*	GCN	GCN(contact)	0.148(0.002)	0.889(0.002)	0.885(0.002)
MGraphDTA (Yang et al., 2022)*	MGNN	MCNN	0.150(0.004)	0.890(0.002)	0.883(0.003)
SMT-DTA (Pei et al., 2022)	Transformer	Transformer	0.154	0.894	-
Our (iEdgeDTA)	Edge-GCN	1D-GCN	0.142(0.004)	0.890(0.001)	0.888(0.001)

* these results were taken from our reproduction.

Module analysis with baseline

Batch normalization

Batch normalization was a valuable addition to our strategy for enhancing model regularization. It's a common technique in deep learning to normalize input data within mini-batches during training. Doing so mitigates the issue of internal covariate shift, which occurs when the distribution of inputs to model layers changes as parameters are updated. This stability aids in faster convergence during training. Furthermore, batch normalization introduces a form of regularization by estimating expectations and variances from smaller data subsets.

The influence of batch normalization is shown in Figure 6.1. In the initial stages of training before the red dashed line, the model incorporating batch normalization achieves convergence more quickly compared to the model without this feature. As training progresses, the mean squared error (MSE) consistently decreases for the model without batch normalization, suggesting an improved fit to the training data. However, the outcomes on the validation set, as shown in Figure 6.2, reveal inferior performance of the model without

batch normalization relative to other models. This outcome implies the presence of overfitting and emphasizes the importance of batch normalization in enhancing the model's ability to generalize.

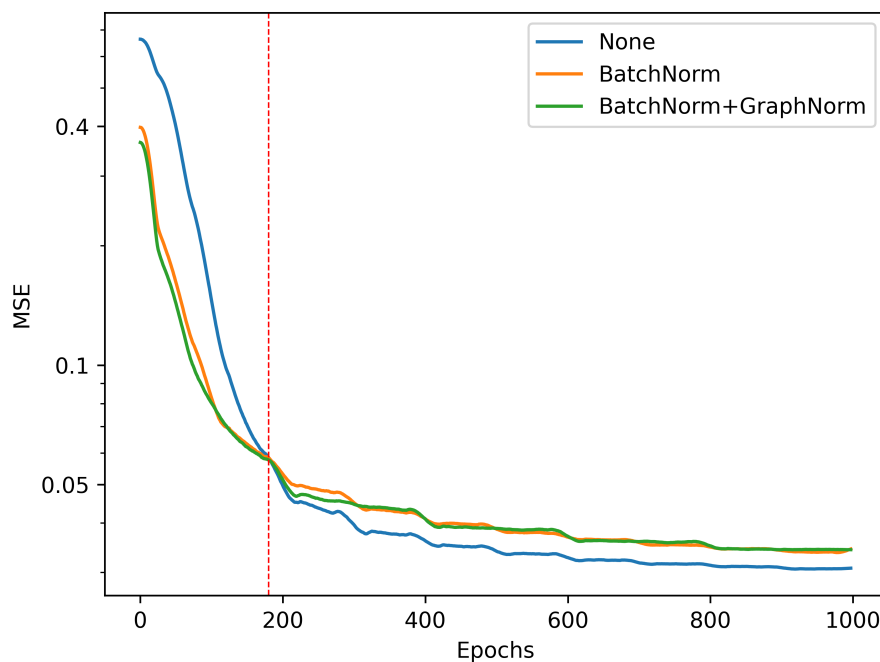


Figure 6.1: The training performance of the model varies across different batch normalization configurations, with the model without batch normalization (depicted in blue) exhibiting the most favorable performance on the training set (lower is better).

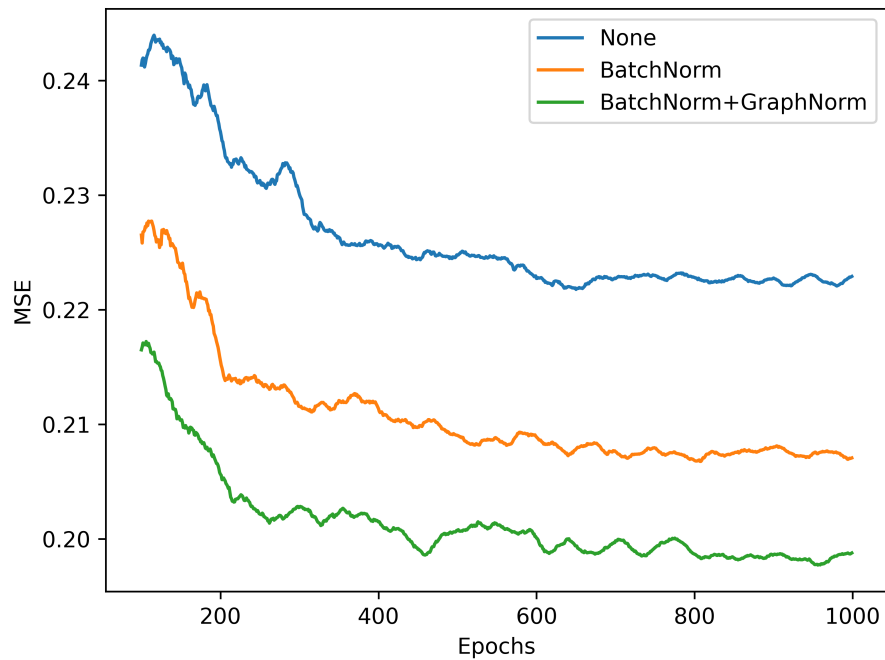


Figure 6.2: The validation performance of the model varies across different batch normalization configurations, the model with both BatchNorm and GraphNorm (green) exhibiting the most favorable performance on the validation set (lower is better).

We experimented to investigate the potential regularization impact of introducing a dropout layer to enhance the model's generalization. As depicted in Figure 6.3, the experiment's findings indicate that increasing the dropout probability decreased the model's performance on the validation set, suggesting that the regularization effect provided by batch normalization is sufficient for our model.

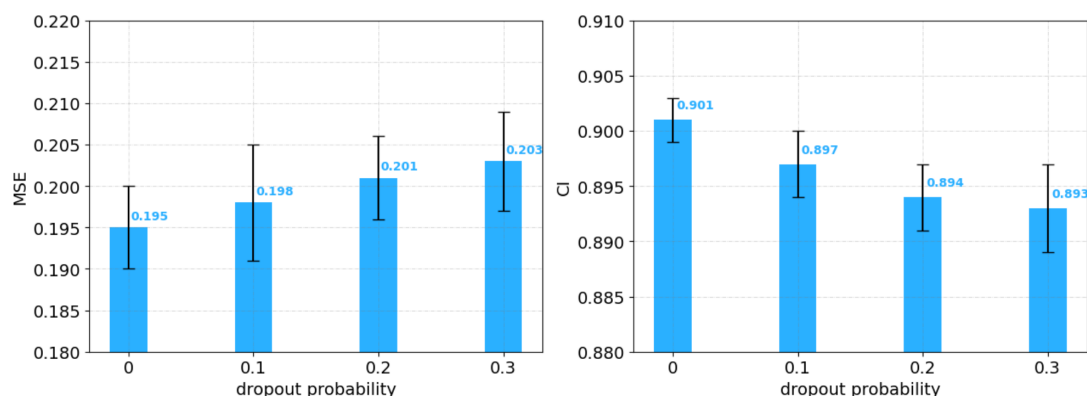


Figure 6.3: The performance of different dropout probabilities is assessed on Davis’s five-fold cross-validation set, with the error bars representing the standard deviation. The left panel presents the averaged mean squared error (MSE), while the right panel displays the averaged concordance index (CI) scores.

Pre-trained model

This research utilizes a graph neural network to analyze proteins using their sequences, assuming that acquiring the real or predicted protein structure is impractical. Therefore, relying on a pre-trained model is dependable due to its comprehensive training on a vast dataset, possibly encompassing implicit information about protein features and structures. As a result, we selected two pre-trained models to generate an input features for our model. We assessed their ability predictive capability using the Davis dataset, as indicated in Tables 6.3 in the Pre-trained choice section.

The findings demonstrated that both pre-trained models outperformed the baseline model, which depended on manually crafted features for protein representation. ESM-1v outperformed ProtT5, getting a lower MSE value of 0.202 and a higher CI value of 0.897. Therefore, we decided to use ESM-1v as our selected pre-trained model for the final assessment.

Edge and Global features

By including edge features in the computation of the Graph Convolutional Network (GCN) through the introduction of an Edge-GCN layer, we are able to increase the representation of the drug sequence. This is achieved by integrating edge features generated from the drug sequence at each iteration of message-passing. However, acquiring the edge features of proteins remains a complex and challenging task. Thus, to obtain a better representation of proteins, an alternative approach was implemented. We created a global protein feature by calculating the mean of its pre-trained node-level features and then fed it into a learning process using three fully connected layers. The effect of incorporating the drug compound's edge features into the model is demonstrated in Table 6.3 in the Edge feature section. The integration of protein global features is evaluated using a final model called iEdgeDTA, as described in the Protein global feature section.

The results revealed that incorporating edge features exclusively from the drug branch enhanced performance, resulting in an MSE value of 0.198 and a CI value of 0.899. Subsequently, we evaluated the contribution of the protein's global feature, demonstrating a positive effect on prediction accuracy. This resulted in a reduction in MSE to 0.195 and an increase in the CI score to 0.901.

Table 6.3: Performance of model component on average five-fold Davis’s cross validation set.

Model	MSE(std)	CI(std)
Baseline (DgraphDTA)	0.212(0.006)	0.885(0.004)
Pre-trained choice		
ProtT5	0.206(0.008)	0.895(0.005)
ESM-1v	0.202(0.007)	0.897(0.005)
Edge feature		
ESM-1v w/o edge feature	0.202(0.007)	0.897(0.005)
ESM-1v w/ edge feature	0.198(0.006)	0.899(0.002)
Protein global feature		
ESM-1v w/ edge feature	0.198(0.006)	0.899(0.002)
iEdgeDTA(ESM+edge+global)	0.195(0.005)	0.901(0.002)

Furthermore, we conducted a comparison between our method, which integrates edge features (Edge-GCN), and existing approaches such as RGCN (Schlichtkrull et al., 2018). RGCN introduces the concept of relation-specific weights for edges in the graph. This means that each type of relationship between nodes is associated with distinct weights, allowing the model to learn and distinguish different types of connections within the graph. In this specific context, relation types correspond to various forms of chemical bonds, such as single bonds, double bonds, triple bonds, aromatics, and self-loops, resulting in a total of five distinct weights. The results, as presented in Table 6.4, reveal that the incorporation of RGCN does not lead to an improvement in model performance during the five-fold cross-validation. This result can be attributed to our approach’s utilization of multidimensional edge features,

whereas RGCN is limited to accommodating only a single feature as the relation type

Table 6.4: Comparison between the edge integration approach using a five-fold Davis’s cross validation set.

Model	Edge feature	MSE(std)
RGCN	relation	0.201(0.006)
Our (GCN)	none	0.201(0.006)
Our (Edge-GCN)	muti-dimension	0.198(0.006)

Moreover, we integrated the Edge-GCN technique into the baseline model, DGraphDTA, to evaluate the effectiveness of including edge information in drug representations. In this experimental configuration, we replaced the original GCN layers in the drug branch with our Edge-GCN layers, incorporating the edge feature proposed in this research paper. The results, presented in Table 6.5, illustrate that incorporating an edge feature into the graph neural network yields enhancements in both MSE and CI scores.

Table 6.5: Comparison between the original DGraphDTA and DGraphDTA with edge integration approach using a five-fold Davis’s cross validation set.

Model	MSE(std)	CI(std)
DGraphDTA	0.212(0.007)	0.888(0.004)
w/ Edge-GCN	0.207(0.006)	0.895(0.002)

Protein node and sequence features analysis

We used a Convolutional Neural Network (CNN) architecture as our initial model to evaluate the influence of node features and sequence features on protein feature extrac-

tion. The selection was made based on its extensive utilization in previous studies and its appropriateness for assessing the efficacy of our 1D-GCN in resolving the fixed-length issue. The CNN hyperparameters were set according to the methodology described in the GraphDTA(Nguyen et al., 2020) publication. More precisely, we ensured that all protein sequences had a consistent length of 1000 amino acids. For shorter sequences, we added extra padding, while for longer sequences, we truncated them to fit the desired length. The results of the experiment are presented in Table 6.6, demonstrating the performance of specific node-level and sequence-level features as independent versions. These studies emphasize the individual gains gained by each feature, highlighting their separate contributions.

Furthermore, the incorporation of both node-level and sequence-level characteristics in the 1D-GCN model resulted in substantial improvements, surpassing the performance of the baseline model. This outcome highlights the benefits of integrating these features, ultimately leading to enhanced precision in predicting affinity scores.

Table 6.6: Contribution of node-Level and sequence-Level features in combination with a CNN baseline model on a five-fold cross-validation set from Davis’s dataset.

Model	Method	MSE(std)	CI(std)
CNN baseline	CNN	0.210(0.007)	0.891(0.003)
Node feature	1D-GCN	0.198(0.005)	0.899(0.002)
Global feature	Linear	0.202(0.005)	0.896(0.003)
iEdgeDTA	1D-GCN+Linear	0.195(0.005)	0.901(0.002)

Contact map performance

Finally, we conducted a further investigation to see how protein structure affects our model. In particular, we replaced a 1D sequence graph with a contact map predicted by different contact map models. We chose three sources for this experiment: pconsc4 (Michel et al., 2018), which is used in our baseline model DGraphDTA; RaptorX (Wang et al., 2017)

used by GEFA; and the ESM (Meier et al., 2021) contact map prediction model, with a contact determination threshold of 0.5, the probability used by these models to indicate the likelihood that a pair of amino acids is in contact.

To conduct this analysis, we ensured a uniform model architecture and maintained constant hyperparameters while making adjustments to the protein graph construction. Furthermore, we incorporated our 1D sequence graph into the DGraphDTA model to evaluate the influence of 1D-GCN within an alternative model setup. The outcomes of this analysis are presented in Table 6.7.

Table 6.7: Comparison of 1D-GCN performance with predicted contact maps from different contact map prediction models on the average five-fold cross-validation set of Davis’s dataset.

Contact map model	MSE(std)	CI(std)
DGraphDTA		
pconsc4	0.212(0.006)	0.885(0.004)
1D-GCN	0.211(0.007)	0.893(0.001)
iEdgeDTA		
pconsc4	0.197(0.005)	0.901(0.003)
RaptorX	0.196(0.006)	0.899(0.004)
ESM	0.198(0.004)	0.900(0.004)
1D-GCN	0.195(0.005)	0.901(0.002)

The results of our study reveal that the performance of the 1D-GCN is comparable to that of a model employing a contact map, as observed in both our iEdgeDTA experiment and the baseline DGraphDTA experiment. These findings suggest that information derived

from contact map predictions may contain noise, potentially affecting model performance. This emphasizes the necessity of having either an actual contact map or exceptionally accurate predictions in order to achieve better performance. However, generating protein contact maps is a challenging and complex operation that requires specialized equipment, experience, and resources. These obstacles hinder their accessibility and restrict them to research facilities.

Our iEdgeDTA model, which relies on only protein sequence, possesses the potential to be effectively utilized in practical applications, particularly in virtual screening. This attribute gives it a valuable asset in scenarios where the majority of the data remains in the sequential format rather than the structural format and the availability of actual contact maps is still limited or requires significant computational resources.

Limitations

Nevertheless, iEdgeDTA does possess specific constraints. Firstly, using a one-dimensional representation has a disadvantage since the folding of the protein determines the position of the binding pocket, and the one-dimensional model does not ensure that the binding pocket aligns with a consecutive sequence of amino acids in the linear string. Therefore, it is still difficult to see or predict the binding pocket merely based on the one-dimensional depiction.

Furthermore, a constraint arises from the utilization of the pre-trained model, which could potentially create a noisy feature when extracting lengthy protein sequences, as evaluated in Algorithm 1 (Chapter 4 Proposed approach). Using a more accurate pre-trained model that can process sequences of any length might reduce its susceptibility to noise, potentially improving its performance.

Conclusions

Accurate prediction of drug-target affinity plays a vital role in identifying promising candidates for new drug development and optimizing drug design. Additionally, it can significantly reduce the time and costs associated with drug discovery by enabling drug repositioning, which assesses whether existing drugs can effectively bind to specific protein targets. In this research, we introduced a graph-based deep learning model that en-

hances prediction performance by incorporating background knowledge, introducing multi-dimensional edge features, and utilizing more complex protein node features derived from pre-trained models. Through a comparative analysis with recent studies employing various methodologies, we demonstrated that our approach outperforms existing methods in terms of prediction accuracy.

Furthermore, we assessed the model's efficacy when incorporating protein structure information based on contact maps. Interestingly, our findings suggested that protein structure is optional for achieving high prediction accuracy in our model. Given that our model currently focuses on integrating edge features into the drug branch, future research could enhance accuracy by improving the extraction of protein node and edge features.



REFERENCES

- Cai, T., Luo, S., Xu, K., He, D., yan Liu, T., and Wang, L. 2020. Graphnorm: A principled approach to accelerating graph neural network training.
- Davis, M. I., Hunt, J. P., Herrgard, S., Ciceri, P., Wodicka, L. M., Pallares, G., Hocker, M., Treiber, D. K., and Zarrinkar, P. P. 2011. Comprehensive analysis of kinase inhibitor selectivity. Nature Biotechnology 29.11 (Nov 2011): 1046–1051.
- Elnaggar, A., Heinzinger, M., Dallago, C., Rehawi, G., Yu, W., Jones, L., Gibbs, T., Feher, T., Angerer, C., Steinegger, M., Bhowmik, D., and Rost, B. 2021. Prottrans: Towards cracking the language of lifes code through self-supervised deep learning and high performance computing. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021): 1–1.
- gensas.org 2023. Sequences tab [Online]. Available from: <https://www.gensas.org/sequences> [2023,December].
- He, T., Heidemeyer, M., Ban, F., Cherkasov, A., and Ester, M. 2017. Simboost: a read-across approach for predicting drug–target binding affinities using gradient boosting machines. Journal of Cheminformatics 9.1 (Apr 2017): 24.
- Ioffe, S. and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, ICML'15, p. 448–456. null: JMLR.org.
- Jarada, T. N., Rokne, J. G., and Alhaji, R. 2020. A review of computational drug repositioning: strategies, approaches, opportunities, challenges, and directions. Journal of Cheminformatics 12.1 (Jul 2020): 46.
- Jiang, M., Li, Z., Zhang, S., Wang, S., Wang, X., Yuan, Q., and Wei, Z. 2020. Drug–target affinity prediction using graph neural network and contact maps. RSC Adv. 10 (2020): 20701–20712.
- Jiang, M., Wang, S., Zhang, S., Zhou, W., Zhang, Y., and Li, Z. 2022. Sequence-based drug–target affinity prediction using weighted graph neural networks. BMC Genomics 23.1 (Jun 2022): 449.

- Khataniar, A., Pathak, U., Rajkhowa, S., and Jha, A. N. 2022. A comprehensive review of drug repurposing strategies against known drug targets of covid-19. COVID 2.2 (2022): 148–167.
- Kipf, T. N. and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In Proceedings of the 5th International Conference on Learning Representations, ICLR '17. null: Proceedings of the 5th International Conference on Learning Representations.
- Landrum, G., Tosco, P., Kelley, B., Ric, sriniker, gedek, Cosgrove, D., Vianello, R., Nadi-Schneider, Kawashima, E., N, D., Dalke, A., Jones, G., Cole, B., Swain, M., Turk, S., AlexanderSavelyev, Vaucher, A., Wójcikowski, M., Take, I., Probst, D., Scalfani, V. F., Ujihara, K., guillaume godin, Pahl, A., Berenger, F., JLVarjo, jasondbiggs, strets123, and JP 2023. rdkit/rdkit: 2022_09_5 (q3 2022) release [Online]. Available from: <https://doi.org/10.5281/zenodo.7671152> [2023,February].
- Loshchilov, I. and Hutter, F. 2017. Decoupled weight decay regularization [Online]. Available from: <https://arxiv.org/abs/1711.05101> [2017,null].
- Meier, J., Rao, R., Verkuil, R., Liu, J., Sercu, T., and Rives, A. Language models enable zero-shot prediction of the effects of mutations on protein function, 2021. Available from: <https://www.biorxiv.org/content/early/2021/11/17/2021.07.09.450648> .
- Michel, M., Menéndez Hurtado, D., and Elofsson, A. 2018. PconsC4: fast, accurate and hassle-free contact predictions. Bioinformatics 35.15 (12 2018): 2677–2679.
- Ng, Y. L., Salim, C. K., and Chu, J. J. H. 2021. Drug repurposing for COVID-19: Approaches, challenges and promising candidates. Pharmacol Ther 228 (June 2021): 107930.
- Nguyen, T., Le, H., Quinn, T. P., Nguyen, T., Le, T. D., and Venkatesh, S. 2020. GraphDTA: predicting drug–target binding affinity with graph neural networks. Bioinformatics 37.8 (10 2020): 1140–1147.
- Nguyen, T. M., Nguyen, T., Le, T. M., and Tran, T. 2022. GEFA: Early fusion approach in Drug-Target affinity prediction. IEEE/ACM Trans Comput Biol Bioinform 19.2 (April 2022): 718–728.

- Pei, Q., Wu, L., Zhu, J., Xia, Y., Xie, S., Qin, T., Liu, H., and Liu, T.-Y. 2022. Smt-dta: Improving drug-target affinity prediction with semi-supervised multi-task training [Online]. Available from: <https://arxiv.org/abs/2206.09818> [2022,null].
- Schlichtkrull, M., Kipf, T. N., Bloem, P., van den Berg, R., Titov, I., and Welling, M. 2018. Modeling relational data with graph convolutional networks. In Gangemi, A., Navigli, R., Vidal, M.-E., Hitzler, P., Troncy, R., Hollink, L., Tordai, A., and Alam, M. (ed.), The Semantic Web, pp. 593–607. Cham: Springer International Publishing.
- sciencesnail.com 2019. The difference between ki, kd, ic50, and ec50 values [Online]. Available from: <https://www.sciencesnail.com/science/the-difference-between-ki-kd-ic50-and-ec50-values> [2019,Dec].
- Shin, B., Park, S., Kang, K., and Ho, J. C. 2019. Self-attention based molecule representation for predicting drug-target interaction. In Doshi-Velez, F., Fackler, J., Jung, K., Kale, D., Ranganath, R., Wallace, B., and Wiens, J. (ed.), Proceedings of the 4th Machine Learning for Healthcare Conference, volume 106 of Proceedings of Machine Learning Research, pp. 230–248. null: PMLR.
- Steinegger, M., Mirdita, M., and Söding, J. 2019. Protein-level assembly increases protein sequence recovery from metagenomic samples manifold. Nat Methods 16.7 (June 2019): 603–606.
- Tang, J., Szwajda, A., Shakyawar, S., Xu, T., Hintsanen, P., Wennerberg, K., and Aittokallio, T. 2014. Making sense of large-scale kinase inhibitor bioactivity data sets: a comparative and integrative analysis. J Chem Inf Model 54.3 (February 2014): 735–743.
- UniProt Consortium. 2007. The universal protein resource (UniProt). Nucleic Acids Res 36.Database issue (November 2007): D190–5.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. 2017. Attention is all you need [Online]. Available from: <https://arxiv.org/abs/1706.03762> [2017,].
- Wang, S., Sun, S., Li, Z., Zhang, R., and Xu, J. 2017. Accurate de novo prediction of protein contact map by ultra-deep learning model. PLOS Computational Biology 13.1 (01 2017): 1–34.

- Weininger, D. 1988. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. Journal of Chemical Information and Computer Sciences 28.1 (1988): 31–36.
- Wikipedia 2023. Simplified molecular-input line-entry system — Wikipedia, the free encyclopedia. <http://en.wikipedia.org/w/index.php?title=Simplified%20molecular-input%20line-entry%20system&oldid=1136187945>.
- Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P. S. 2021. A comprehensive survey on graph neural networks. IEEE Transactions on Neural Networks and Learning Systems 32.1 (jan 2021): 4–24.
- Xu, K., Hu, W., Leskovec, J., and Jegelka, S. 2018. How powerful are graph neural networks? [Online]. Available from: <https://arxiv.org/abs/1810.00826> [2018,].
- Yang, Z., Zhong, W., Zhao, L., and Yu-Chian Chen, C. 2022. Mgraphdta: deep multi-scale graph neural network for explainable drug–target binding affinity prediction. Chem. Sci. 13 (2022): 816–833.
- Zhu, Z., Shi, C., Zhang, Z., Liu, S., Xu, M., Yuan, X., Zhang, Y., Chen, J., Cai, H., Lu, J., Ma, C., Liu, R., Xhonneux, L.-P., Qu, M., and Tang, J. 2022. Torchdrug: A powerful and flexible machine learning platform for drug discovery. null (2022):
- Öztürk, H., Özgür, A., and Ozkirimli, E. 2018. DeepDTA: deep drug–target binding affinity prediction. Bioinformatics 34.17 (09 2018): i821–i829.

Biography

Natchanon suviriyapaisal was born in May, 1997. He graduated from Khonkaen University where he received B.Eng in mechanical engineering. Currently, he is working on his master degree in Computer Engineering from Chulalongkorn University. His field of interested includes applied machine learning and deep learning.

