GTA-S : Graph Transformer for binding affinity using atomic diStance

Léna N. Ezzine
lena.ezzine@student.ecp.fr

Théo Moutakanni
theo.moutakanni@student.ecp.fr

Omar Souaidi
omar.souaidi@student.ecp.fr

Oussama Boussif
oussama.boussif@student.ecp.fr

Abstract

Drug discovery is an important domain in the medical field and drug laboratories have been able to battle many diseases like covid for example, and it makes use of Drug-Target binding Affinity (**DTA**) prediction in order to design drugs with certain properties. In this report we try to predict whether a drug (ligand) can bind to a target (protein) by modeling them as graphs in addition to considering the distance between atoms.

1 Introduction

Binding affinity is the strength of the binding interaction between a single biomolecule (e.g. protein or DNA) to its ligand/binding partner (e.g. drug or inhibitor). It is typically measured and reported by the equilibrium dissociation constant (K_D), which is used to evaluate and rank order strengths of bimolecular interactions. The smaller the K_D value, the greater the binding affinity of the ligand for its target. The larger the K_D value, the more weakly the target molecule and ligand are attracted to and bind to one another.

Predicting accurate binding affinity between a small molecule and target protein is one of the fundamental challenges in drug development. Drug discovery is one of the first steps needed to enable a successful drug product to get to market and due to the complex nature of molecule discovery, there is a high risk of failure, when compounds do not behave as expected, do not have the required activity, or show issues during development.

According to [3], the most common approach to measuring affinity is to vary the concentration of one component, while keeping the concentration of the other binding partner constant. However, this experimental design is not always sufficient, as there are two limiting regimes, determined by the concentration of the constant component; only one of these concentration regimes allows the K_D to be reliably determined, while the other does not and this can lead to errors with orders of magnitude from the true constant K_D.

In recent years, deep learning has known great advances and many architectures emerged such as CNNs for images and transformers that were first used in NLP but most importantly, Graph Neural Networks (GNN) that enabled models to learn on graph structured data. First methods involved using machine learning for predicting DTA, in [1] authors used SVM on Property Encoded Shape Distributions (PESD) signatures. These signatures are invariant to rotation and translation and incorporate three dimensional distributions of mapped properties on a molecular surface. This allows to take distance into account but the interaction between individual atoms isn't strongly captured. In [7] the authors used the outer-product of column vectors of Tanimoto similarity matrix and Smith–Waterman similarity matrix for the drugs and targets, respectively and fed them to 2D CNNs, however this approach doesn't take into account relative distance between atoms and doesn't fully make use of the inductive bias for graphs. In the same spirit, authors of [4] used an ensemble of

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

3D-CNN networks to predict DTA, this approach makes use of the positional information, however it needs a larger dataset in order to obtain good results. Authors of [6], used GNNs to predict drug-target binding affinity which makes uses not only of atoms but also the covalent bonds, however, as pointed out in [9], the spatial information of relative atom positions must be taken into account. The S-MAN architecture [9] uses positional encodings that incorporates the distance between atoms and creates two graphs, one for node-edge interaction and another for edge-edge interaction and feed them to a graph attention network [8].

Our work is organized as follows: In section 4 we explain our approach, in section 5 we present our experimental setup and report its results in section 6. Finally, we perform an ablation study in section 7.

2 Dataset

We use PDPBind 2016 dataset pre-processed by S-MAN's code and the training, validation and testing split is the same as S-MAN.

Dataset comprises of 3390 graphs for the training set, 377 for validation and 290 for testing.

3 Preliminaries

3.1 Problem Statement

Let us clarify some concepts of DTA prediction:

- The drug is referred to as a ligand in DTA prediction.
- The protein is referred to as target.

Given a drug compound and a target protein, the DTA prediction task is to predict the binding affinity between them. In general, we use L and P to represent the input drug (or ligand) and the input target (or protein) **separately**. Both can be a graph, a sequence, or other format input. The predicted binding affinity y is a continuous real number value, obtained with the following regression task :

$$f : (L, P) \Rightarrow y \tag{1}$$

The limitation of splitting the drug and target in 2 inputs is that we don't make use of the information of distances between them. Hence, a more appropriate formulation is to represent drug-protein complex as protein-ligand binding pose with preserving the essential spatial structure, we call it pocket-ligand. The pocket-ligand graph can be denoted as G , and the distance matrix of atoms in graph G can be represented as D. The construction of G and D will be introduced next. Now the problem can be defined as:

$$g : (G, D) \Rightarrow y \tag{2}$$

3.2 Pocket-Ligand graph construction

As we claimed, the structure information in the original molecular graph with only the covalent bonding correlation is not enough. More spatial edges should be included in the graph to provide adequate 3D structure information. What's more, there is no natural bonding between ligand and protein. Therefore, the input interaction graph of our proposed model is the spatial-enhanced pocket-ligand graph (SPoG). We denote the SPoG by a new graph $G = (V = V_M \cup V_P, E)$, where $V = \{a_1, a_2, \ldots, a_N\}$, V_M and V_P are the atom set of ligand and protein pocket. To build spatial-enhanced edges for G , we first calculate the spatial distances between all atoms in 3D space, the distance matrix is denoted as D. Then a threshold θ_d is set to preserve the correlation edge e_{ij} between a pair of atoms. In this way, the edge set can be built:

$$E = \{e_{ij} = (v_i, v_j) | v_i, v_j \in V, D_{ij} \leq \theta_d\} \tag{3}$$

4 Methodology

4.1 Method 1 - Graph Transformers With Edges Features

In this section, we replace all the S-MAN architecture with the Graphs Transformer Architecture.

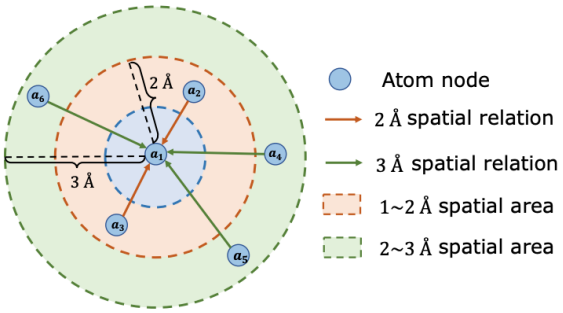

Figure 1: One hot distance encoding example

4.1.1 Data preprocessing

First, we proceed with the pocket-ligand graph construction, as stated in section 2.2.

Each pocket-ligand is characterized by the following:

- Number of nodes (atoms),
- Edges list,
- Nodes features,
- 3D coordinates of each node.

4.1.2 Edges features initialisation

As stated in last part, now we have our initial node features. How do we initialise edge features ? Given an edge $Edge_{ij}$, it would be relevant to take into account :

- Features $\mathbf{a_i}$ and $\mathbf{a_j} \in \mathbb{R}^z$ of the corresponding nodes $node_i$ and $node_j$ that form the edge,
- The distance between these 2 nodes.

In our case, we don't take directly the scalar distance between 2 nodes. Instead, we use the same distance encoding as S-MAN.

The position of atoms in pocket-ligand is defined by 3D coordinates, forming the input position matrix $S \in \mathbb{R}^{N\times 3}$. Considering the variability of coordinates that are manually defined, we convert S into a relative spatial matrix, that is distance matrix $D \in \mathbb{R}^{N\times N}$. From Figure 2)b) from [9], we noticed that spatial distance indicates different meaningful correlations between atoms. Therefore, the spatial information is encoded by applying a one-hot encoder to split the scalar distances into b buckets, leading to a multiple position relation matrix $D^R \in \mathbb{R}^{N\times N\times b}$. Taking fig. 1 as an example, we divide neighbors of a_1 into different spatial relations. This gives us, for each edge $Edge_{ij}$, an encoded distance $d_{ij} \in \mathbb{R}^d$. The final edge feature e_{ij} is represented as :

$$e_{ij} = [\mathbf{a_i} \oplus \mathbf{a_j} \oplus d_{ij}] \tag{4}$$

Where $\oplus$ is the concatenation operation over two vectors and $e_{ij} \in \mathbb{R}^y$ where $y = z + z + d$.

4.1.3 Laplacian Positional Encoding

We pre-compute the Laplacian eigenvectors of all graphs in the dataset. Eigenvectors are defined via the factorization of the graph Laplacian matrix :

$$L = I - D^{-1/2}AD^{-1/2} = U^T VU \tag{5}$$

where A is the $n \times n$ adjacency matrix, D is the degree matrix, and V, U correspond to the eigenvalues and eigenvectors respectively. We use the k smallest non-trivial eigenvectors of a node as its positional encoding. We denote by $\lambda_i \in \mathbb{R}^k$ the obtained representation for node i.

4.1.4 Graph Transformers architecture

We take the graph transformers architecture from [2]. The process is illustrated in fig. 2

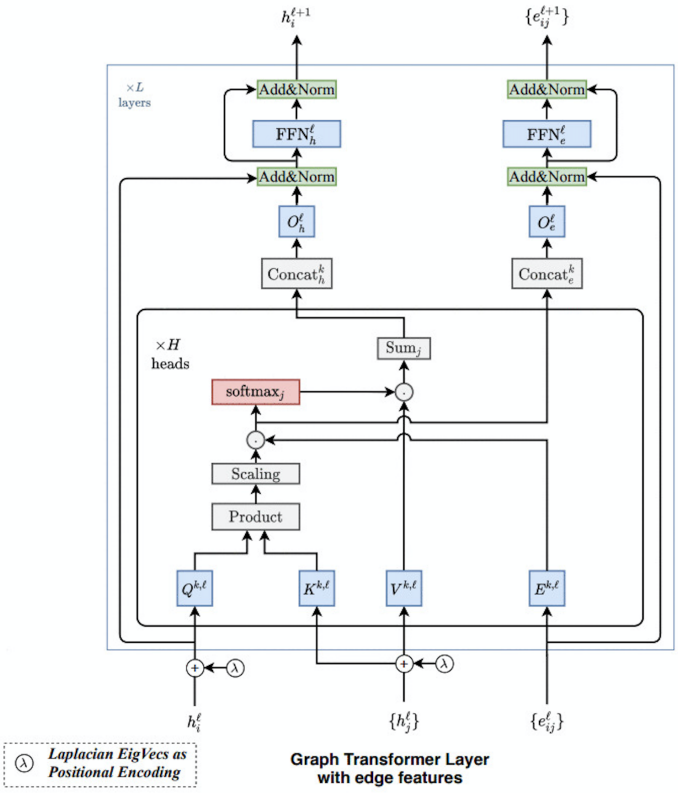

Figure 2: Block Diagram of Graph Transformer Layer with edge features

Input First we embed the node features and edges features to d-dimensional hidden features h_i^0 and e_{ij}^0.

$$h_i^0 = A^0\mathbf{a_i} + a^0 \quad ; \quad e_{ij}^0 = B^0 e_{ij} + b^0 \tag{6}$$

Where $A^0 \in \mathbb{R}^{d\times z}$, $B^0 \in \mathbb{R}^{d\times y}$ and $a^0, b^0 \in \mathbb{R}^d$ are the parameters of the linear projection layers. We embed then the pre-computed node positional encodings of dim k via a linear projection and add to the hidden node features h_i^0.

$$\hat{\lambda}_i = C^0\lambda_i + c^0 \quad ; \quad h_{ij}^0 = h_{ij}^0 + \hat{\lambda}_i \tag{7}$$

Where $C^0 \in \mathbb{R}^{d\times k}$ and $c^0 \in \mathbb{R}^d$.

Graph Transformer Layer With Edges Features For this part, we refer to the original paper [2] in section **2.3** as it is the same.

Task based MLP Layers The output of the previous section is the graph G with final node features $h_i, i \in \{1, .., num_{nodes}\}$. For the MLP, we refer to the original paper [2] in section **2.3** as it is the same. Let us highlight that, before passing the graph to the MLP, there are many ways to aggregate the node features into one final representation : we could take the sum of $h_i, i \in \{1, .., num_{nodes}\}$, the maximum, or the mean.

Loss For our regression task, we use the mean-squared error loss between the prediction and true value of the binding affinity.

4.2 Method 2 - Distance-aware Molecule Graph Attention Network

We tried to reproduce the results of [9]. The model is best described in the original paper.

We used the same data pre-processing as in the method 1. We tried to use both the original spatial features and the new Laplacian positional encoding as edges features.

The code was done using the dgl library.

Two main differences exist between our code and the original one :

- First, to allow different hidden sizes between network layers, we added two encoders on Eq. 10 to encode the node and edge features instead of only one that is shared.
- Secondly, to represent bidirectional graphs, we used parallel unidirectional edges. This is due to the library we used but double the number of edge features while reducing the training time by a factor of 2-4x.

5 Experimental Setup

We trained our model on 300 epochs using batch size of 64, using the AdamW optimizer [5] along with a polynomial decay scheduler with warmup [1] that decreases the learning rate from the initial learning rate $\mathbf{3.10^{-4}}$ to the ending learning rate $\mathbf{5.10^{-7}}$ after a warmup period corresponding to 65 epochs (3500 iterations) which it increases linearly from 0 to the initial learning rate. In order to evaluate the model on the test set, we consider the checkpoint's model that performed best on the validation set.
After some finetuning (see the ablation study section 7), we come up with the the model's parameters that are summarized in Table 1.

Parameters	1^{st} method	2^{nd} method
Positional encoding dimension	20	20
Hidden dimension of nodes and edges in the Graph Layers	256	128
Number of attention heads	8	8
Num. of Graph Transformer Layers	4	4
Aggregation function of the nodes right before the MLP	Sum	Sum

Table 1: Baseline model parameters

These 2 heuristics improved our results, hence we will keep them for our upcoming experiments.

6 Results

Now, let us look at the results obtained with the S-MAN architecture, compared to the results obtained with our architecture for the first method and second method. For each model, we did 6 runs but we only performed one run for the second method. Here, we report the mean and the standard deviation.

Model	Val MSE	Test MSE
S-MAN	1.39±0.02	1.38 ±0.02
Graph Transformers - 1^{st} method	**1.25 ±0.03**	**1.30 ±0.02**
Graph Transformers - 2^{nd} method	**1.39**	**1.54**

Table 2: Results obtained on the PDPBind Dataset, with S-MAN and the Graph Transformers architecture

The graph Transformer architecture outperformed the S-MAN on predicting the binding affinity for the PDPBind dataset, both for the validation and the test sets. We can explain this with the following reasons :

- Graph transformers in the first method are an improvement of GAT (used in S-MAN), because Graph transformers closes the gap between the original Transformer, specifically by using the multi-headed self-attention mechanism, and the graph neural network.
- In each graph transformes layer, the nodes and edges hidden features are updated **together at once**. This was not the case for S-MAN, in which we first update the edge hidden features, then the node hidden features. This architecture specifically brings exciting promise to datasets where domain information along pairwise interactions can be leveraged for maximum learning performance.

[1] https://huggingface.co/transformers/_modules/transformers/optimization.html#get_polynomial_decay_schedule_wit_warmup

- The second method doesn't perform as well and overfits quickly (around 100 epochs) and this might be due to the fact that the model is way too big for the number of datapoints in the dataset

7 Ablation study

In this section, we only present the ablation study for the first method since it was the one that gave good results.

7.1 Impact of Positional encoding dimension

In this part, we explore the impact of the positional encoding dimension on the performances of the model. Since the smallest molecule in our dataset has 22 atoms, we choose 20 as the largest dimension, and experiment lower dimensions. Results are shown in table 3.

Pos.Enc. Dimension	Val MSE	Test MSE
6	1.35	1.35
10	1.26	1.30
15	1.23	1.28
20	**1.22**	**1.27**

Table 3: Impact of Positional encoding dimension

We notice that the dimension of the nodes positional encoding has a significant impact on the results : the bigger the dimension, the better the results. This can be explained by the fact that higher-dimensional positional encoding carry more information about the graph structure.

7.2 Impact of the number of Graph Transformer layers

We experiment changing the number of graph transformer layers in the model architecture. Results are shown in table 4.

Num. Graph transformer layers	Val MSE	Test MSE
3	1.23	1.34
4	**1.22**	**1.27**
5	1.25	1.36

Table 4: Impact of the number of graph transformer layers

The best result is obtained with 4 layers, on both the validation and the test sets. Our hypothesis to explain this is that 3 layers is not enough to capture all the dataset patterns, and 5 layers leads to overfitting because there are too many trainable parameters.

7.3 Impact of the hidden dimension in the Graph layers

In this part, we experiment changing the dimension of the hidden representation h of nodes and edges in the graph transformer layers. Results are show in table 5.

Hidden dim.	Val MSE	Test MSE
128	1.25	1.38
256	**1.22**	**1.27**

Table 5: Impact of the hidden dimension of nodes and edges in Graph Transformer Layers

We can see that the best result is obtained with dimension 256 . This means that 128 is not enough to capture the complexity of representations of our data.

8 Conclusion

In this work, we build upon the S-MAN architecture that incorporates distance information by leveraging the power of graph transformer networks in place of the attention layers and the use of laplace embeddings. Moreover, our approach only makes use of one graph instead of creating two

graphs, one for edge-edge interaction and the other for node-edge interaction. As a consequence, our model performs better than S-MAN and is less complex that S-MAN.

References

[1] Sourav Das, Michael Krein, and Curt Breneman. "Binding Affinity Prediction with Property-Encoded Shape Distribution Signatures". In: *Journal of chemical information and modeling* 50 (Feb. 2010), pp. 298–308. DOI: 10.1021/ci9004139.

[2] Vijay Prakash Dwivedi and Xavier Bresson. "A Generalization of Transformer Networks to Graphs". In: *arXiv preprint arXiv:2012.09699* (2020).

[3] Inga Jarmoskaite et al. "How to measure and evaluate binding affinities". In: *eLife* 9 (Aug. 2020). DOI: 10.7554/elife.57264. URL: https://doi.org/10.7554/elife.57264.

[4] Yongbeom Kwon et al. "AK-Score: Accurate Protein-Ligand Binding Affinity Prediction Using an Ensemble of 3D-Convolutional Neural Networks". In: *International Journal of Molecular Sciences* 21 (Nov. 2020), p. 8424. DOI: 10.3390/ijms21228424.

[5] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. 2019. arXiv: 1711.05101 [cs.LG].

[6] Thin Nguyen et al. "GraphDTA: Predicting drug–target binding affinity with graph neural networks". In: *Bioinformatics* (Oct. 2020). btaa921. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btaa921.

[7] Jooyong Shim et al. "Prediction of drug–target binding affinity using similarity-based convolutional neural network". In: *Scientific Reports* 11 (Feb. 2021), p. 4416. DOI: 10.1038/s41598-021-83679-y.

[8] Petar Veličković et al. *Graph Attention Networks*. 2017. eprint: arXiv:1710.10903.

[9] Jingbo Zhou et al. *Distance-aware Molecule Graph Attention Network for Drug-Target Binding Affinity Prediction*. 2020. eprint: arXiv:2012.09624.