Drug-drug Interaction Prediction with Learnable Size-Adaptive Molecular Substructures

Journal:	*Briefings in Bioinformatics*
Manuscript ID	BIB-21-1051.R1
Manuscript Type:	Problem solving protocol
Date Submitted by the Author:	12-Sep-2021
Complete List of Authors:	Nyamabo, Arnold Kazadi; Northwestern Polytechnical University, YU, Hui; Northwestern Polytechnical University, Liu, Zun; Northwestern Polytechnical University, Computer Science Shi, Jianyu; Northwestern Polytechnical University, School of Life Sciences
Keywords:	Drug-drug interaction, Substructure extraction, Substructure interaction, Multi-type interactions, Gated Message Passing Neural Network
Note: The following files were submitted by the author for peer review, but cannot be converted to PDF. You must view these files (e.g. movies) online.	
oup-authoring-template.tex	

Journal Title Here, 2019, pp. 1–10

doi: DOI HERE
Advance Access Publication Date: Day Month Year
Paper

PAPER

Drug-Drug Interaction Prediction with Learnable Size-Adaptive Molecular Substructures

Arnold K. Nyamabo,[1] Hui Yu,[1,*] Zun Liu[1] and Jian-Yu Shi[2,*]

[1]School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China and [2]School of Life Sciences, Northwestern Polytechnical University, Xi'an 710072, China
[*]Corresponding authors: Hui Yu huiyu@nwpu.edu.cn, Jian-Yu Shi jianyushi@nwpu.edu.cn

FOR PUBLISHER ONLY Received on Date Month Year; revised on Date Month Year; accepted on Date Month Year

Abstract

Drug-Drug Interactions (DDIs) are interactions with adverse effects on the body, manifested when two or more incompatible drugs are taken together. They can be caused by the chemical compositions of the drugs involved. We introduce gated message passing neural network (GMPNN), a message passing neural network which learns chemical substructures with different sizes and shapes from the molecular graph representations of drugs for DDI prediction between a pair of drugs. In GMPNN, edges are considered as gates which control the flow of message passing, and therefore delimiting the substructures in a learnable way. The final DDI prediction between a drug pair is based on the interactions between pairs of their (learned) substructures, each pair weighted by a relevance score to the final DDI prediction output. Our proposed method GMPNN-CS (i.e., GMPNN + prediction module) is evaluated on two real-world datasets, with competitive results on one, and improved performance on the other compared with previous methods. Source code is freely available at `https://github.com/kanz76/GMPNN-CS`.

Key words: Drug-Drug Interaction, Substructure extraction, Substructure interaction, Multi-type interactions, Gated Message Passing Neural Network

1. Introduction

Drug-Drug Interactions (DDIs) refer to the phenomenon, with adverse effects on an organism, triggered when two or more drugs are taken together. The co-administration of more than one drug can be justified by the fact that some diseases are just too complex to be treated with a single drug [1], or that multiple diseases warrant multiple medications [2]. DDIs are therefore the risks, sometimes life-threatening [2], that come with the therapeutic benefits sought in multiple medications. The assessment of these risks have prompted many studies and research work aimed at identifying whether two or more given drugs are safe to be taken together.

The identification of DDIs is usually performed in pharmaceutical research/setting through extensive experimental testings (*in vitro*) and clinical trials. However, the huge number of combinations of drugs that should be considered for experimental testings makes this process highly expensive and quasi-impossible, even with high-throughput methods [3]. Computational methods (*in silico*) can thus be used as a cheap, yet effective and fast alternative to alleviate this problem by predicting potential DDIs based on the knowledge distilled from already known DDIs.

In the last few years, there has been significant progress with exciting results from the many learning based methods (i.e., both machine and deep learning methods) proposed for the identification of potential DDIs between combinations of *two* drugs, also known as DDI prediction task. However, most of these methods are limited in the way that they represent drugs as inputs, and perform the DDI prediction task.

Based on the medicinal chemistry knowledge [4], which states that a drug is simply an entity composed of different functional groups/chemical substructures which determine all of its pharmacokinetic (how it is handled by an organism) and pharmacodynamic (how it affects an orgnanism) proprieties, and ultimately all of its interactions, we propose a simple, yet competitive computational method for DDI prediction (including specifying the type of the interaction) between *two* drugs equipped with this inductive bias. *First*, a drug is represented as a graph based on its molecular graph representation. In order to extract the substructures of a drug from its graph, we propose a message passing neural network where edges have learnable weights, constrained within the interval [0, 1]. These weights can be considered as gates to delimit the substructures, with the effect of producing flexible-sized and irregular-shaped substructures. *Second*, the DDI prediction between two drugs is based on the interaction scores between their learned substructures, each one weighted by a learnable weight using a co-attention mechanism [5] (or interaction map). As a byproduct, our method can give a hint at what substructures might be the cause of a DDI occurrence. *Third*, we evaluate our model on two real-word datasets: (1) Drugbank dataset which contains 1,704 drugs, 191,808 DDI pairs and 86 interaction types. (2) Twosides dataset with 645 drugs, 4,649,441 DDI pairs, and 1,317 interaction types. Experiments are conducted under two settings: transductive setting, where the training and test sets of DDI pairs share the same drugs; and inductive setting (also referred to as cold-start), where drugs in the two sets have to be different. The latter is more challenging than the former. Both datasets are used in the transductive setting where our method showed competitive results on DrugBank dataset, and outperformed previous methods on the Twosides dataset. In the inductive scenario, only the DrugBank dataset is considered, and our method shows better results.

2. Related Work

This section discusses previous works in DDI prediction task from two perspectives: (1) How drugs are represented, and (2) how the actual prediction is performed.

2.1. Drug Representation

The vast majority of existing DDI prediction methods represent drugs using molecular fingerprints [6–8], and/or other drug profiles such as side effects [6, 7], binding targets [8], transporters, enzymes, pathways, or the combination of two or more of these features [7, 9, 10]. Molecular fingerprints [11, 12] are binary vectors whose elements indicate the presence (1) or absence (0) of a specific chemical substructure. The other profiles are similarly represented as binary vectors indicating the presence or absence of a particular profile, say, a specific side effect or binding target. Some methods [13–18] perform even further preprocessing by representing a drug as a similarity vector which indicates how similar it is to other drugs in the aforementioned representation spaces using similarity measures such as cosine similarity, Jaccard similiarty. This is guided by the assumption that similar/dissimilar drugs are likely to have similar/dissimilar biological activities [17]. The downside with these representations is that they are hand-crafted, limited to the current state of human knowledge. There is not enough flexibility to discover beyond what is encoded in them by a domain expert. For instance, with fingerprint representation, there is no way to discover beyond the chemical substructures that are already predefined, especially when dealing with new drugs. Additionally, some features such as side effects are not always available, especially in early drug development, and therefore impeding the methods that rely on them to be used.

In recent years, graph neural networks (GNNs) [19–22], deep learning models designed for graph-structured data, have been applied for learnable task-tailored representations of chemical molecules in general, and drugs in particular, with improved performance in molecule-related tasks [23–26]. However, this is usually optimized to learn the representation of a drug as a whole entity without giving too much consideration to the principal actors of DDIs, that is the functional groups/chemical substructures that constitute the drug molecule. Some of recent methods [27, 28] are proposed with the consideration of substructures involvement in DDIs. However, nodes' hidden representations (also referred to as patch representations) at each GNN layer are direclty treated as substructure representations of the drugs. This approach produces substructures of regular shapes whose diameters/sizes are defined by the receptive field of the GNN layer. In this work, in order to extract substructures from a molecular graph, we propose a method that directly learns substructures of different sizes and shapes of the molecule.

2.2. Drug-drug interaction prediction

Approaches for DDI prediction can roughly be classified into two categories: one category where the drugs form a graph or network, and the other where they are considered independent from one another. In the latter, in order to perform the DDI prediction between two drugs, their representations are aggregated (e.g., summation, concatenation) and then fed into a linear or non-linear classifier for prediction [10, 13, 29–31]. In the network-based category, drugs are assumed to form an interconnected system where the drugs are the nodes, and the edges can either represent the DDIs between the nodes [7, 14, 16, 17, 32, 33] or the similarities between the drugs, based on their representations [6, 18]. Different graph specific methods, including label propagation [6], matrix factorization [7, 8, 16], graph auto-encoders [18, 33] are applied to these derived networks to either conduct the prediction or firstly learn low-dimensional representations of the drugs and then perform the DDI prediction. The advantage of network-based methods is the addition of the drug interconnected system's topological information to the drug representations which can boost the performance. However, this approach does not work in inductive setting. In this work, our method considers drugs as independent entities, and therefore can be used both in inductive and transductive settings. Furthermore, we leverage the co-attention mechanism (same as in [27, 30]) between the learned substructures of a drug pair so that each drug can communicate to the other which substructures are really relevant. This has the effect that drugs are not completely handled independently.

3. Method

In this section, we mathematically formulate the problem we are trying to solve and present the building blocks of our method, including the input format and all involved computational steps. The overall framework is illustrated in Fig. 1.

3.1. Problem Formulation

Given a set of drugs $\mathcal{D}$ and a set of interaction types $\mathcal{R}$, the DDI prediction task can be regarded as a function $f : \mathcal{D}\times\mathcal{R}\times\mathcal{D} \rightarrow [0, 1]$. That is, given a triplet of two drugs and an interaction of a certain type, the task predicts the probability that this type of interaction will occur between the two drugs. The goal is to find an approximation of f, given a dataset of known DDIs $\mathcal{M} = \{(d_x, r, d_y)_i\}_{i=1}^{N} \subset \mathcal{D} \times \mathcal{R} \times \mathcal{D}$. The proposed method in this section is one such approximation.

3.2. Inputs

Drugs are represented as hydrogen-depleted undirected graphs $G = (V, E)$, where V is the set of nodes, representing atoms; $E \subset V \times V$ are the edges, representing the (covalent) bonds between the atoms. Each node v_i has a corresponding feature vector $\mathbf{x}_i \in \mathbb{R}^d$. Similarly, each edge $e_{ij} = (v_i, v_j)$ has a feature vector $\mathbf{x}_{ij} \in \mathbb{R}^{d'}$. The features used for atoms and bonds are given in the Supplementary

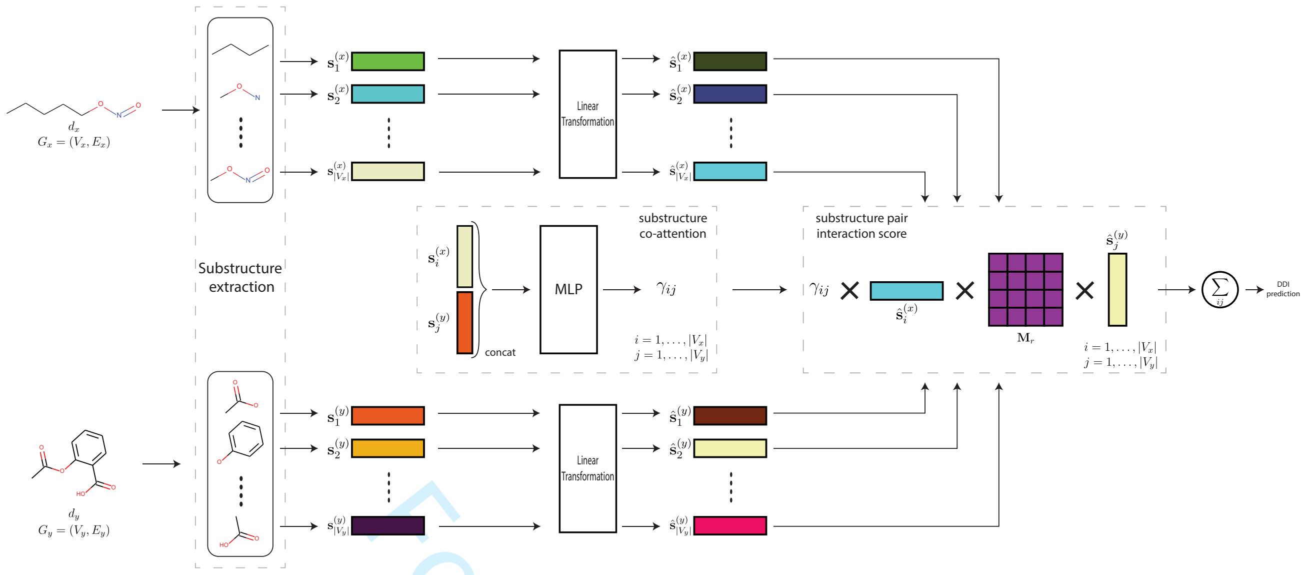


Fig. 1. Overview of our method workflow. Given a DDI tuple (d_x, r, d_y), d_x and d_y are represented in their molecular graph representation G_x and G_y, respectively, and r is represented as the learnable matrix $\mathbf{M_r} \in \mathbb{R}^{b\times b}$. After substructure extraction (Section 3.3), G_x produces $|V_x|$ $(\{\mathbf{s}_i^{(x)}\}_{i=1}^{|V_x|})$ substructures where each node is the center of a substructure. G_y undergoes the same process. A cross-substructure attention γ_{ij} is computed between these extracted substructures to learn how relevant they are to one another (Eq. 22). Each substructure undergoes a linear transformation, and the final DDI prediction is the sum of interaction scores between every pair of drugs d_x and d_y's substructures, each one weighted by the co-attention weight γ_{ij} (Eq. 17).

material Section 1. Note that at this initial stage, because the graph is undirected, the edges e_{ij} and e_{ji} are practically the same, and so are $\mathbf{x}_{ij}$ and $\mathbf{x}_{ji}$.

3.3. Substructure extraction with gated message passing neural network

Given a general graph (i.e., not only a molecular graph), the representation of a substructure centered[1] at a node v_i can be regarded as the aggregation of all the nodes $\{v_j \mid v_j \in \mathcal{P}_{\to i}\}$ on all the paths (starting at end nodes[2]) in the graph that end at v_i. $\mathcal{P}_{\to i}$ means a path to node v_i. To refine the substructure even further, edges on the graphs can be considered as gates, that is, with weight values constrained within [0, 1]. These gates control the flow of information along a path. A node v_j along a path to v_i is therefore weighted by the product of the weight values of the edges along the path that connect it to v_i. We assume that if there are many paths from v_j to v_i, which can happen if there is a cycle, each path is considered separately. Weighing a node by the product of edge values as proposed here has the end effect of producing substructures of different sizes and shapes. If along the path, an edge has weight value of ≈ 0, the nodes on the rest of the path are cut off, therefore, delimiting the substructure in an irregular shape. Each node in the graph can be treated as the center of a substructure in order to extract as many as possible substructures from the graph. However, this will require that each edge be transformed into a bi-directional edge (each direction with its own weight) because a node can be both a target node v_i (i.e., center) and a source node v_j.

[1] The node v_i is not literally at the center, it is simply the end node of all the paths.

[2] These can considered as peripheral nodes or nodes that are as far away as possible from the node being considered as the center.

See Fig. 2 for an illustrative example of the overall substructure extraction process.

Applying this concept to a drug for chemical substructure extraction, its molecular graphical representation $G = (V, E)$, which initially is undirected, is converted into a directed graph. Edges e_{ij} and e_{ji} become two separate edges. To highlight this difference, they are renamed $e_{i\to j}$ (edge from node v_i to node v_j) and $e_{j\to i}$, respectively. Each (directed) edge is assigned a learnable weight constrained within the range 0-1.

The generation of all the paths in the graph can be computationally inefficient, we therefore propose a message passing neural network (MPNN) [21] named gated message passing neural network (GMPNN) which can simulate this process. MPNN is a framework of multi-layer spatial convolutional GNNs. Each layer comprises three main components, namely, message passing (Eq. 1), aggregation (Eq. 2), and update (Eq. 3):

$$\mathbf{m}_{j\to i}^{(l)} = M^{(l)}\left(\mathbf{h}_j^{(l-1)}, \mathbf{h}_i^{(l-1)}, \mathbf{q}_{j\to i}\right), \quad \forall j : v_j \in \mathcal{N}(v_i) \qquad (1)$$

$$\mathbf{a}_i^{(l)} = A^{(l)}\left(\{\mathbf{m}_{j\to i}^{(l)}\}_{j:v_j\in\mathcal{N}(v_i)}\right) \qquad (2)$$

$$\mathbf{h}_i^{(l)} = U^{(l)}\left(\mathbf{h}_i^{(l-1)}, \mathbf{a}_i^{(l)}\right) \qquad (3)$$

At the l^{th} iteration/layer $(l = 1, \ldots, L)$, in order to get the updated feature vector $\mathbf{h}_i^{(l)}$ of v_i, Eq. 1 flows information/message from its adjacent nodes $(\mathcal{N}(v_i))$ v_j's feature vectors $\mathbf{h}_j^{(l-1)}$ of the previous iteration $(l-1)$. This information can be constrained by or conditioned on node's v_i previous feature vector $\mathbf{h}_i^{(l-1)}$, and the features $\mathbf{q}_{j\to i}$, if any, of the edge from v_j and v_i, with the possibility of applying a non-linear transformation $M^{(l)}$. In Eq. 2, all these messages are gathered using an aggregation function $A^{(l)}$ (e.g., sum, max, mean) to finally produce a newly updated feature vector $\mathbf{h}_i^{(l)}$ of v_i in Eq. 3. $U^{(l)}$ is an update function which can be as simple as the sum of its arguments, or a complex non-linear function.. In short, at each iteration, a node is passed messages (features) from

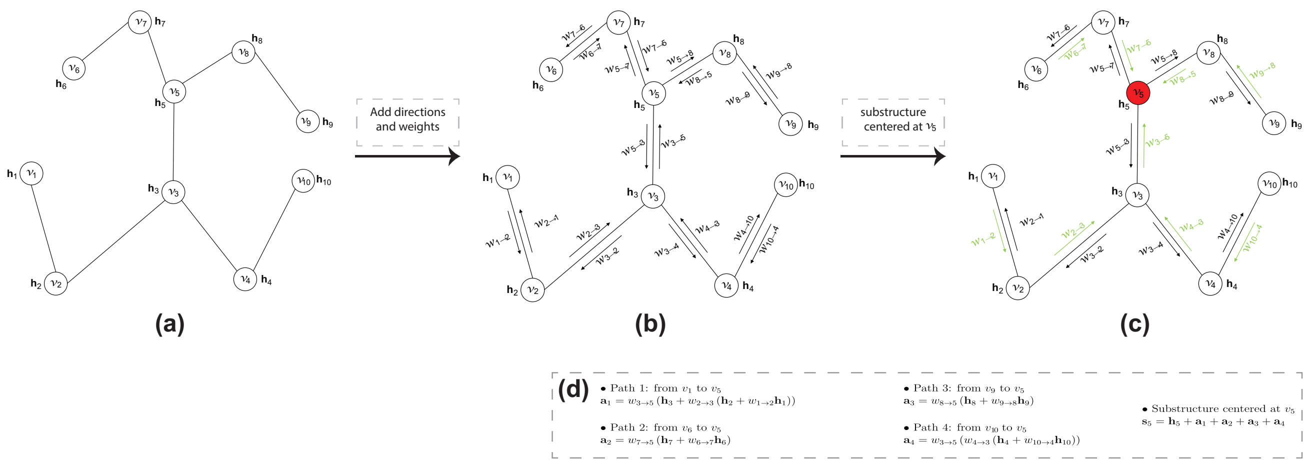

(a)
Add directions and weights
(b)
substructure centered at v_5
(c)
(d)
• Path 1: from v_1 to v_5 $\mathbf{a}_1 = w_{3\to5}(\mathbf{h}_3 + w_{2\to3}(\mathbf{h}_2 + w_{1\to2}\mathbf{h}_1))$
• Path 2: from v_6 to v_5 $\mathbf{a}_2 = w_{7\to5}(\mathbf{h}_7 + w_{6\to7}\mathbf{h}_6)$
• Path 3: from v_9 to v_5 $\mathbf{a}_3 = w_{8\to5}(\mathbf{h}_8 + w_{9\to8}\mathbf{h}_9)$
• Path 4: from v_{10} to v_5 $\mathbf{a}_4 = w_{3\to5}(w_{4\to3}(\mathbf{h}_4 + w_{10\to4}\mathbf{h}_{10}))$
• Substructure centered at v_5 $\mathbf{s}_5 = \mathbf{h}_5 + \mathbf{a}_1 + \mathbf{a}_2 + \mathbf{a}_3 + \mathbf{a}_4$

Fig. 2. Example of substructure extraction on **(a)** a toy graph, v_i represents a node with associated feature vector $\mathbf{h}_i$. **(b)** Every edge is transformed into a bi-directional one with weights $w_{j\to i}$ within [0, 1] for each direction. **(c)** In order to extract the substructure centered at node v_5 (in red), all the paths (shown in green color) starting at end-nodes and ending at v_5 are generated. **(d)** The substructure centered at v_5 represented by vector $\mathbf{s}_5$ is the aggregation of all the nodes along these paths, each node scaled by the product of the weights of the edges linking it to v_5.

its adjacent nodes. This has the effect that at iteration l, a node would be updated with the features of all the nodes that can be reached within a walk of length l. This is close to our idea of node aggregation along paths for substructure extraction. A walk in a graph has a redundancy in the nodes visited (a node can appear more than once in a walk, but at most once in a path), to get even closer to the path generation process, we will base our work on a MPNN variant, proposed in [26], named directed message passing neural network (D-MPNN). Here, in order to reduce the redundancy of nodes with standard MPNN, messages are passed between edges instead of nodes. The three components of MPNN in D-MPNN become:

$$\mathbf{m}_{k\to j}^{(l)} = M^{(l)}\left(\mathbf{h}_k, \mathbf{h}_j, \mathbf{q}_{k\to j}^{(l-1)}\right), \quad \forall k : v_k \in \mathcal{N}(v_j) \setminus \{v_i\} \tag{4}$$

$$\mathbf{a}_i^{(l)} = A^{(l)}\left(\{\mathbf{m}_{k\to j}^{(l)}\}_{k:v_k\in\mathcal{N}(v_j)\setminus\{v_i\}}\right) \tag{5}$$

$$\mathbf{q}_{j\to i}^{(l)} = U^{(l)}\left(\mathbf{q}_{j\to i}^{(l-1)}, \mathbf{a}_{j\to i}^{(l)}\right) \tag{6}$$

The difference between MPNN and D-MPNN is that the former updates node features, while the latter updates edge features. In order to update edge $e_{j\to i}$, Eq. 4 passes messages from its neighboring edges $e_{k\to j}$, where the node v_j is the common vertex. Care is taken to remove the opposite direction to $e_{j\to i}$, that is, the edge $e_{i\to j}$ (hence, $\mathcal{N}(v_j) \setminus \{v_i\}$ in Eq. 4), so that information flows only in one direction, therefore reducing redundancy. After the final iteration L of edge features updates, nodes are represented as the aggregation of the features of all their incoming edges. For instance, node v_i's final feature representation $\mathbf{s}_i$ can be:

$$\mathbf{s}_i = \sum_{j:v_j\in\mathcal{N}(v_i)} \mathbf{q}_{j\to i}^{(L)} \tag{7}$$

Next, we present the computational steps of our message passing method GMPNN. Given a graph $G = (V, E)$ representing a drug:

- A non-linear transformation is applied to the nodes for better feature representation:

$$\mathbf{h}_i = \text{MLP}_{init\_n}(\mathbf{x}_i), \quad \forall v_i \in V \tag{8}$$

where $MLP_{init\_n}(\cdot)$ is a multi-layer perceptron (MLP) for non-linear transformation. $\mathbf{h}_i \in \mathbb{R}^f$ is the updated feature vector of node v_i after transforming its original feature vector $\mathbf{x}_i$.

- The edge features are also transformed as follows:

$$\mathbf{h}_{j\to i} = \text{MLP}_{init\_e}(\mathbf{x}_{ji}), \quad \forall e_{j\to i} \tag{9}$$

$$\mathbf{h}_{i\to j} = \text{MLP}_{init\_e}(\mathbf{x}_{ij}), \quad \forall e_{i\to j} \tag{10}$$

$\mathbf{h}_{j\to i}, \mathbf{h}_{i\to j} \in \mathbb{R}^m$ are new feature vectors of edges $e_{j\to i}$ and $e_{i\to j}$, respectively. Note that even though the edges $e_{j\to i}$ and $e_{i\to j}$ are different, their features $\mathbf{h}_{j\to i}$ and $\mathbf{h}_{i\to j}$ are practically the same because $\mathbf{x}_{ji} = \mathbf{x}_{ij}$ (See Section 3.2).

- The weight (gate) $w_{j\to i} \in [0, 1]$ of edge $e_{j\to i}$ is initialized based on its incident nodes' features $\mathbf{h}_j$ and $\mathbf{h}_i$, and its own features $\mathbf{h}_{j\to i}$:

$$o_{j\to i} = \frac{1}{c}\left(\mathbf{h}_{j\to i}^T \text{MLP}_w(\mathbf{h}_j \parallel \mathbf{h}_i)\right)$$

$$w_{j\to i} = \sigma(o_{j\to i}) \tag{11}$$

where $\sigma(\cdot)$ is the sigmoid function used to constrain $w_{j\to i}$ within [0, 1]; ^{T} is the transposition operation; $\parallel$ is the concatenation operation. c (> 0) is a constant used to avoid saturation with gradient flow when using sigmoid function. We have set it to the degree of node v_j, the tail of the edge $e_{j\to i}$. Note that the edges $e_{j\to i}$ and $e_{i\to j}$ have different weights (i.e., $w_{j\to i} \neq w_{i\to j}$), because $\parallel$ is not commutative.

- Now that we have node feature vectors $\mathbf{h}_i$ and edge weights $w_{j\to i}$, we are ready to apply the directed message passing mechanism for the simulation of aggregation of nodes along paths for substructure extraction. Since message is passed between edges instead of nodes, we propose to promote nodes' features to edge level. Edge $e_{j\to i}$'s new features become:

$$\mathbf{q}_{j\to i}^{(0)} = w_{j\to i}\mathbf{h}_j \tag{12}$$

where $\mathbf{q}_{j\to i}^{(0)} \in \mathbb{R}^f$. That is, $e_{j\to i}$ takes on the features $\mathbf{h}_j$ of its tail node v_j scaled by its weight $w_{j\to i}$. Note that $e_{j\to i}$ previous features $\mathbf{h}_{j\to i}$ (Eq. 9) is different from $\mathbf{q}_{j\to i}^{(0)}$. The former was simply used to compute the weight $w_{j\to i}$ (Eq. 11), whereas the latter (containing node information) will be used in the actual message passing for node aggregation along paths as intended.

- The message passing, aggregation, and update components of our proposed method are therefore defined as:

$$\mathbf{m}^{(l)}_{k\to j} = w_{j\to i}\mathbf{q}^{(l-1)}_{k\to j}, \quad \forall k : v_k \in \mathcal{N}(v_j) \setminus \{v_i\} \tag{13}$$

$$\mathbf{a}^{(l)}_{j\to i} = \sum_{k:v_k\in\mathcal{N}(v_j)\setminus\{v_i\}} \mathbf{m}^{(l)}_{k\to j} \tag{14}$$

$$\mathbf{q}^{(l)}_{j\to i} = \mathbf{q}^{(0)}_{j\to i} + \mathbf{a}^{(l)}_{j\to i} \tag{15}$$

Contrary to Eq. 4, Eq. 13 does not apply any transformations to feature vectors at each iteration l during message passing because we want to keep all the nodes that compose a substructure in the same feature space. However, at each iteration, features are scaled by the weight of the edge, this has the end effect of multiplying node features by the product of the weights of the edges linking it to the center node of the substructure. As an aside, this paradigm of not applying any transformations during message passing can also be regarded as an instance of the concept of simplifying graph convolutions [34].

- The final representation of a node v_i, after the last iteration L, which captures the substructure information of which it is the center is given by:

$$\mathbf{s}_i = f_{sub}\left(\mathbf{h}_i + \sum_{j:v_j\in\mathcal{N}(v_i)} \mathbf{q}^{(L)}_{j\to i}\right) \tag{16}$$

where $\mathcal{N}(v_i)$ is the set of nodes adjacent to v_i, and $f_{sub}(\cdot)$ is a non-linear function implemented as an MLP. To obtain $\mathbf{s}_i \in \mathbb{R}^b$, we simply aggregate all the features $\mathbf{q}^{(L)}_{j\to i}$ from all of its incoming edges $e_{j\to i}, \forall j : v_j \in \mathcal{N}(v_i)$. $\mathbf{s}_i$ is the vector representation of the learned/extracted substructure centered at v_i. Note that initially, the feature vector of node v_i (i.e., $\mathbf{h}_i$) contained only the information of a single atom, but now $\mathbf{s}_i$ contains the features representing a substructure centered at that atom.

The application of our proposed method for substructure extraction in inductive setting is made possible by the fact that the values of edge weights (Eq. 11) depend only on node (atom) and edge (bond) features. If we encounter a new drug, the substructure extraction operation can still be performed because all the molecules share the same type of atoms and bonds.

3.4. Drug-drug interaction prediction

Given a DDI tuple (d_x, r, d_y), the DDI prediction is determined as the join probability of the tuple:

$$P(d_x, r, d_y) = \sigma\left(\sum_{i}^{|V_x|}\sum_{j}^{|V_y|} \gamma_{ij}\hat{\mathbf{s}}^{(x)T}_i \mathbf{M}_r \hat{\mathbf{s}}^{(y)}_j\right) \tag{17}$$

- $\sigma(\cdot)$ is the sigmoid function.
- $\hat{\mathbf{s}}^{(x)}_i$ and $\hat{\mathbf{s}}^{(y)}_j$ are linear transformations of the substructure $\mathbf{s}^{(x)}_i$ and $\mathbf{s}^{(y)}_j$, respectively.

$$\hat{\mathbf{s}}^{(x)}_i = \mathbf{W}_{(x)}\mathbf{s}^{(x)}_i, \qquad i = 1, \dots, |V_x| \tag{18}$$

$$\hat{\mathbf{s}}^{(y)}_j = \mathbf{W}_{(y)}\mathbf{s}^{(y)}_j, \qquad j = 1, \dots, |V_y| \tag{19}$$

where $\mathbf{W}_{(x)}$, and $\mathbf{W}_{(y)} \in \mathbb{R}^{b\times b}$ are learnable transformation matrices.

- $\mathbf{M}_r \in \mathbb{R}^{b\times b}$ is the learnable representation matrix of the interaction type of r. To reduce the number of parameters, we constrain it to be a diagonal matrix.

$$\mathbf{M}_r = \mathrm{diag}(\mathbf{m}_r) \tag{20}$$

diag$(\cdot)$ generates a diagonal matrix where $\cdot$ is the diagonal, and $\mathbf{m}_r \in \mathbb{R}^b$ is a learnable vector specific to the type of interaction r.
Because the Twosides dataset defines multiple existing interactions between two given drugs (See Section 4.1 for more details), we redefine $\mathbf{M}_r$ as follows for this dataset:

$$\mathbf{M}_r = \mathrm{diag}(f_{pred}(\mathbf{m}_r)) \tag{21}$$

where f_{pred} is a non-linear function implemented as an MLP to encourage similar or commonly co-occuring interaction types to have similar represenatitons.

- $\gamma_{ij} \in [0, 1]$ is the cross-substructure interaction weight, also known as co-attention, between substructures (Eq. 16) $\mathbf{s}^{(x)}_i$ of drug d_x and $\mathbf{s}^{(y)}_j$ of d_y.

$$\gamma_{ij} = \mathrm{softmax}\left(\mathrm{MLP}_\gamma(\mathbf{s}^{(x)}_i \parallel \mathbf{s}^{(y)}_j)\right), \quad i = 1, \dots, |V_x| \quad j = 1, \dots, |V_y| \tag{22}$$

Here again to account for the multiplicity of interactions between two drugs in the Twosides dataset, for this dataset we propose :

$$\gamma_{ij} = \mathrm{softmax}\left(\mathrm{MLP}_\gamma\left(\mathbf{s}^{(x)}_i \parallel \mathbf{s}^{(y)}_j \parallel f_r(\mathbf{m}_r)\right)\right) \tag{23}$$

where f_r is an MLP and $\mathbf{m}_r$ is the same as in Eq. 21. The goal is to have a co-attention score aware of the interaction type which is being considered.

The DDI prediction can therefore be considered as a binary prediction of a DDI tuple. Since only known DDIs are given in the dataset $\mathcal{M}$ (See Section 3.1), they are considered as positive samples, negative samples are generated by corrupting either d_x or d_y. That is, given a known DDI tuple (d_x, r, d_y), its derived negative sample is generated by either replacing d_x or d_y. We follow the strategy proposed in [35] for negative sample generation. The learning process of the whole model is done by minimizing the binary cross-entropy loss function given as:

$$\mathcal{L} = -\frac{1}{|\mathcal{M}|}\sum_{i:(d_x,r,d_y)_i\in\mathcal{M}} (\log(p_i) + \log(1 - p'_i)) \tag{24}$$

$|\mathcal{M}|$ is the number of DDI tuples in the dataset, p_i is the probability (Eq. 17) of a known DDI tuple, and p'_i is the probability of its associated negative sample.

4. Experiments

4.1. Datasets

Two real-world datasets, downloaded using the tdc python library[3] [36], were used for the evaluation of our method.

- **DrugBank**: sourced from FDA/Health Canada drug labels, it contains 191,808 DDI tuples with 1,706 drugs. Each drug is represented in SMILES from which molecular graphical

[3] `https://tdcommons.ai/`

representations are generated using the python library RDKit[4]. There are 86 interaction types describing how one drug affects the metabolism of another one. For example, *the excretion of Acamprosate can be decreased when combined with Acetylsalicylic acid (Aspirin)*. Each DDI pair is considered as a positive sample from which a negative sample was generated as mentioned in Section 3.4. In this dataset, there is only one interaction for each DDI tuple, that is, there are no two distinct tuples with the same drug pair but different interactions.

- **Twosides**: proposed by [37] after filtering the original TWOSIDES side effets data [2]. It contains 4,649,441 DDI triplets with 645 drugs, and 1,317 interaction types. As opposed to the DrugBank dataset, these interactions are rather at the phenotypic level than metabolic. That is, here, interactions are simply side effects, such as *headache*, *pain in throat*. Furthermore, given two drugs, there can exist many such interactions between them, contrary to DDI tuples in DrugBank. As in [37], this dataset is further preprocessed by removing interaction types that occur in less than 500 DDI tuples in order to work only with commonly occurring types, thus, remaining with 963 interactions types, and 4,576,287 DDI tuples.

The distributions of the DDI types in both datasets are given in the Supplementary materials Section 4.

4.2. Setup

Our model, named GMPNN-CS, was implemented in Pytorch[5] [38] and Pytorch Geometric[6] [39]. We used random search for hyper-parameters fine-tuning and decided on the best values based on the overall performance on validation set. We considered the following hyper-parameter settings. The number of message passing iterations L was searched from $\{3, 5, 7, 10, 15\}$, dimensions of $\mathbf{h}_i$ (Eq. 8) and $\mathbf{s}_i$ (Eq. 16) were searched from $\{64, 128\}$. The model was trained on mini-batches of 512 DDI tuples using the Adam optimizer [40] with a learning lr rate tuned from {1e-2, 1e-3, 1e-4}. Additionally, an exponentially decaying scheduler of 0.96^t (where t is the current epoch) was set on the learning rate. We found that the combination of $L = 10$, $\mathbf{h}_i \in \mathbb{R}^{64}$, $\mathbf{s}_i \in \mathbb{R}^{128}$, and lr = 1e-3 produced the best performance. The implementation details of MLP_{init-n}, MLP_{init_e}, MLP_w, f_{sub}, MLP_γ, f_r, and f_{pred} are given in the Supplementary materials Section 2.

4.2.1. Baselines

We compared our model with state-of-the-art methods, which similarly (1) work with molecular graphs as inputs; (2) integrate joint drug-drug information in some way during the learning process; (3) consider the involvement of substructures in DDI interaction prediction; and/or (4) work both in transductive and inductive settings.

- **MR-GNN** [28]: uses the representation at each graph convolution layer of nodes to capture substructures of different size for each drug. These representations are jointly fed into a recurrent neural network for a joint representation of a pair of drugs for DDI prediction.
- **MHCADDI** [30]: uses co-attention mechanism to integrate joint drug-drug information during the representation learning of individual drugs.
- **SSI-DDI** [27]: considers each node hidden features as substructures and then computes interactions between these substructures to determine the final DDI prediction.
- **GAT-DDI**: baseline we implemented using graph attention networks (GAT) [19] for drug representations which are directly used for DDI prediction.
- **GMPNN-U**: variant of our proposed method GMPNN-CS where the co-attention coefficient $\gamma_{ij}(Eq.\ 22)$ is simply uniform, that is, $\gamma_{ij} = (|V_x||V_y|)^{-1}$.

We (re-)implemented these methods in Pytorch, some with little modifications from the original work for fair comparison and better performance. See Supplementary material Section 3 for details.

4.3. Results

Experimental results are presented in following metrics: the accuracy (ACC), the area under the receiver operating characteristic (AUC), the average precision (AP), the F1 score, the precision (P), and the recall (R).

4.3.1. Transductive setting

In transductive setting, the drugs used during training can also appear in the test set. In this setting, we split the datasets randomly based on DDI tuples. We performed a stratified split on both datasets based on the interaction types in order to keep the same proportions of interaction types in the training (60% of the data), validation (20%), and test (20%) sets. We did this three times, resulting in three stratified randomized folds. For each DDI tuple, a negative sample is generated as discussed in Section 3.4. They were generated before training to ensure that all the methods are trained on the same data. Each model, including our proposed method and all the baseline models, were trained and tested on each one of these three stratified folds. The means and standard deviations of the results of each model from these three experiments are reported in Table 1. Improvement in performance in each metric score (without considering standard deviations) of our method GMPNN-CS with respect to the highest score of the baseline methods is shown in the bottom row of the table. On the DrugBank dataset, our method did not perform the best as we can see a decrease in scores. We can also see that the results of GMPNN-CS and its variant GMPNN-U are very close, giving the impression that co-attention did not work. An explanation for this behavior is given in Section 5. However, on the Twosides dataset there is an improvement with a significant margin over the other methods. Here, there is a big difference between GMPNN-U and GMPNN-CS because this is a dataset with multiple interactions between a pair of drugs. Due to computational limitation, we could not perform experiments with MHCADDI on this dataset, we only report the results (only AUC available) from the original paper. GAT-DDI does not seem to work well on this dataset, behaving just like a random classifier with scores of about 50%. This can be due to gradient vanishing/exploding or oversmoothing problem. It is not considered in the improvement computation (last row of Table 1) on this dataset. Furthermore, GMPNN-U is similar to GAT-DDI, except for the message passing component. The former uses our proposed GMPNN, whereas the latter uses GAT. Experimental results show that our message passing method performs better than GAT on both datasets. Furthermore, performance on each DDI type on both datasets is presented the Supplementary materials Section 5.

4.3.2. Inductive setting

Contrary to transductive setting, here the dataset is split based on the drugs. That is, the DDI tuples in training and test

[4] `https://www.rdkit.org/`

[5] `https://pytorch.org/`

[6] `https://pytorch-geometric.readthedocs.io/`

Table 1. Comparative evaluation (mean ± std) in % in transductive setting. Best performance in each metric is shown in bold font. The last row shows the improvement in performance on the DDI prediction task in each metric by our method. It is the difference between our method's performance and the best performance among baseline methods. Negative values, therefore, mean our method did not perform the best, and positive ones indicate it performed the best. *GAT-DDI was not considered in the computation.

	DrugBank					Twosides				
	ACC	**AUC**	**AP**	**P**	**R**	**ACC**	**AUC**	**AP**	**P**	**R**
MR-GNN	96.04±0.05	98.87±0.04	**98.57±0.06**	94.48±0.08	**97.78±0.03**	76.23±0.23	85.00±0.22	84.32±0.35	72.82±0.44	83.70±0.39
MHCADDI	83.80±0.27	91.16±0.31	89.26±0.37	78.90±0.06	92.26±0.63	–	88.20	–	–	–
SSI-DDI	**96.33±0.09**	**98.95±0.08**	98.57±0.14	**95.09±0.08**	97.70±0.14	78.20±0.14	85.85±0.13	82.71±0.14	74.33±0.21	86.15±0.15
GAT-DDI	89.81±1.00	95.21±0.70	93.56±0.90	87.04±1.11	93.56±0.52	50.00	50.00	50.00	50.00	100
GMPNN-U	95.00±0.10	98.32±0.04	97.77±0.06	93.19±0.15	97.07±0.06	74.78±0.04	82.08±0.02	78.67±0.03	71.77±0.09	81.69±0.33
GMPNN-CS(Ours)	95.30±0.05	98.46±0.01	97.94±0.02	93.60±0.07	97.22±0.1	**82.83±0.14**	**90.07±0.12**	**87.24±0.12**	**78.42±0.11**	**90.61±0.23**
Improvement	-1.03	-0.49	-0.63	-1.49	-0.56	+4.63	+1.87	+2.92	+4.09	+4.46*

Table 2. Comparative evaluation (mean ± std) in % in inductive setting. Best performance in each metric is shown in bold font.

	ACC	**AUC**	**AP**	**F1**
	$\mathcal{M}_{S1}$ Partition (new drug ↔ new drug)			
MR-GNN	62.63±0.77	70.92±0.84	73.01±1.23	45.81±2.51
MHCADDI	66.50±0.62	72.53±0.92	71.06±1.61	67.21±0.59
SSI-DDI	65.40±1.30	73.43±1.81	75.03±1.42	54.12±3.46
GAT-DDI	66.31±0.61	72.75±0.78	71.61±1.00	**68.68±0.60**
GMPNN-U	67.90±0.50	73.76±0.90	74.58±1.06	63.73±0.80
GMPNN-CS(Ours)	**68.57±0.30**	**74.96±0.40**	**75.44±0.50**	65.32±0.23
	$\mathcal{M}_{S2}$ Partition (new drug ↔ old drug)			
MR-GNN	74.67±0.33	83.15±0.60	83.81±0.69	69.88±0.86
MHCADDI	70.58±0.94	77.84±1.08	76.16±1.45	72.74±0.65
SSI-DDI	76.38±0.92	84.23±1.05	**84.94±0.76**	73.54±1.50
GAT-DDI	69.83±1.41	77.29±1.63	75.79±1.95	73.01±0.85
GMPNN-U	77.00±0.60	83.92±0.50	84.14±0.74	77.29±0.50
GMPNN-CS(Ours)	**77.72±0.30**	**84.84±0.15**	84.87±0.40	**78.29±0.16**

sets do not have overlapping drugs. This approximates a real-world scenario where there is a new drug for which there is no known prior associated drug interactions. It is also referred to as cold-start scenario in the literature. It is more challenging than the transductive setting. In the latter, the model only learns to generalize to unseen DDI tuples (with all the drugs already known during training), while here, the model has to further learn to generalize to unseen drugs (possibly with out-of distribution chemical structures). In this setting, we randomly reserve $\frac{1}{5}$ of the drugs as new drugs ($\mathcal{D}_{new}$), not used during training. DDI tuples whose both drugs are in $\mathcal{D}_{new}$ are collected in $\mathcal{M}_{s1}$, those with neither drugs in $\mathcal{D}_{new}$ are collected in $\mathcal{M}_{train}$, and the remaining tuples, those with one drug in $\mathcal{D}_{new}$ and the other not in, are collected in $\mathcal{M}_{s2}$. This process is also repeated three times, resulting in three randomized folds. Negative samples are generated respecting the constraints of this setting. That is, negative samples for $\mathcal{M}_{train}$ do not contain drugs appearing in $\mathcal{D}_{new}$, those for $\mathcal{M}_{s1}$ have both drugs in $\mathcal{D}_{new}$, and those for $\mathcal{M}_{s2}$ have one and only one drug in $\mathcal{D}_{new}$.
Only DrugBank dataset is used for experiments in this setting because it contains a relatively large number of drugs. We added some regularization in terms of dropout layers [41] to our model so that it does not overfit to old drugs ($\mathcal{D}_{old} = \mathcal{D} \setminus \mathcal{D}_{new}$). To be fair, we also modified the baseline methods by adding dropout layers. Means and standard deviations of the results of experiments on the three randomized folds are shown in Table 2, we can see that our method performed the best overall. Nevertheless, we can observe a significant drop in performance from transductive setting for all the methods. This is because DrugBank is made of drugs that are significantly different as far as their scaffolds (core chemical structure) are concerned [27]. Thus, not only drugs in $\mathcal{D}_{new}$ and $\mathcal{D}_{old}$ are different, but also share very little structurally.

4.4. Visual Inspection

Here are some visual examples of DDI prediction on DrugBank, which can give hints at what might be the cause of a DDI. First, pairs of substructures with top values of γ_{ij} (Eq. 22) are retrieved. Second, the weights of the edges of a substructure in the rendered figures are redefined as follows: for instance, if v_1 is the center of the substructure, and $v_1 \xleftarrow{w_{2,1}} v_2 \xleftarrow{w_{3,2}} v_3 \xleftarrow{w_{4,3}} v_4$ is one of the paths constituting the substructure, where $w_{j,i} (= w_{j \to i}$ (Eq. 11)) is the learned weight of edge $e_{j \to i}$, in the visual examples (Fig. 3-4), edges have values of the products of all the edge values before them in the path and their own values, that is, $v_1 \xleftarrow{w_{2,1}} v_2 \xleftarrow{w_{2,1} w_{3,2}} v_3 \xleftarrow{w_{2,1} w_{3,2} w_{4,3}} v_4$. If $w_{2,1} = 0.9$, $w_{3,2} = 0.7$, and $w_{4,3} = 0.6$, then we have $v_1 \xleftarrow{0.9} v_2 \xleftarrow{0.63 = 0.9 \times 0.7} v_3 \xleftarrow{0.378 = 0.9 \times 0.7 \times 0.6} v_4$. For simplicity, if an edge has more than one neighboring edges (i.e., appears in more than one path), we take the maximum value. Third, these values are shown on corresponding edges and used as intensities to highlight (in green) edges in the figures, and the center nodes of substructures are highlighted with gray-filled circles. Fig. 3 shows the drug sildenafil, a phosphodies-terase-5 (PDE5) inhibitor used for erectile dysfunction, and amyl nitrite, a nitrate drug. It is contraindicated to take a PDE5 inhibitor with a nitrate drug simultaneously because it can cause a reduction in the blood pressure [29, 42]. We can see that nitrate group (in the dashed red box) of Amyl nitrite is very much involved in the DDI prediction outcome. Fig. 4 is the example of ofloxacin (a fluoroquinolone) and calcium carbonate (an antacid). The carbonyl oxygen and the carboxylic acid of ofloxacin (both shown within red dashed boxes) can form a chelate with a metal ion (in this case the calcium of the calcium carbonate). Chelates have very poor water solubility and can therefore cause a significant reduction in the absorption of ofloxacin in the body [4].

5. Discussion and Limitations

Every node/atom is considered the center of a substructure, and therefore there are as many substructures as there are nodes. Nodes that are adjacent to each other end up being centers to substructures that are similar, causing redundancy. This has the negative effect that γ_{ij} (Eq. 22) is overused by only a group of very similar substructures, with the tendency of the former to be uniform within groups of (adjacent) substructures. This can explain why, especially in the case of Drugbank, there is no much difference between GMPNN-U and GMPNN-CS (See Section 4.3.1). As future work, we think that, to solve this issue, a clustering or pooling algorithm might be used to pool similar substructures together and retain only one representative substructure.

The importance of edge weights as gates for substructure extraction only becomes apparent when we increase the number of iterations L. Fig. 5 shows the comparison between GMPNN-CS and a variant (NoGMPNN-CS) where we remove the weights altogether

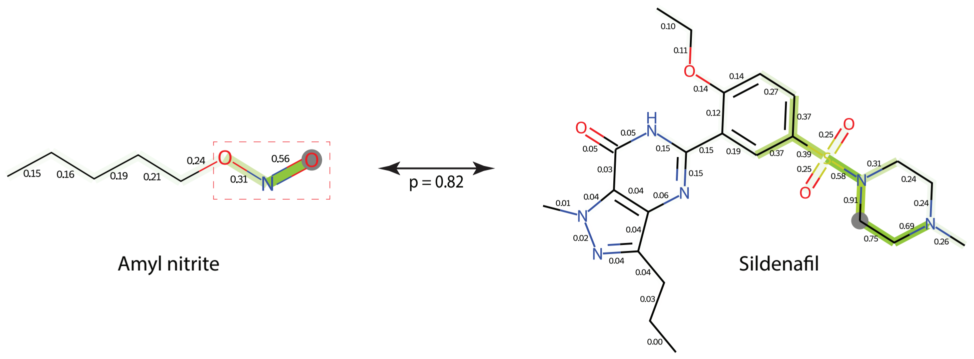


Fig. 3. Visual inspection of DDI prediction between amyl nitrite and sildenafil. $p = 0.82$ is the prediction output. Atoms with gray-filled circles backgrounds are centers of substructures of interest.

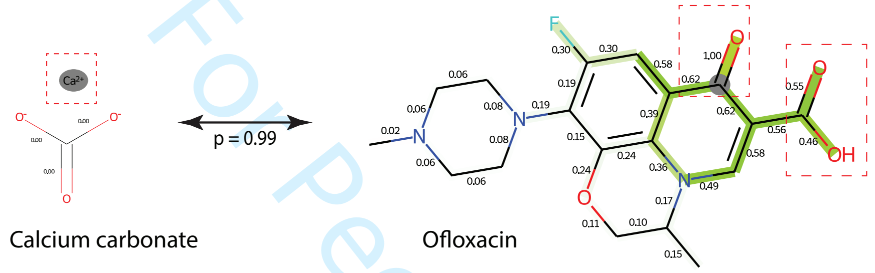


Fig. 4. Visual inspection of DDI prediction between calcium carbonate and ofloxacin.

(or equivalently, the weights are all set to 1). We can see that with 5-10 iterations of message passing the difference is quite small, but becomes significant as the number of iterations increases. As an aside, these results also demonstrate the ability of our proposed method to be extended into deeper GNNs without degradation in performance. Additionally, drugs are mainly organic molecules with the majority of atoms being carbon, producing graphs where the majority of nodes are alike. This can affect the computation of edge weights (Eq. 11). For future work, we think additional information such as spatial location of atoms might be useful to make the difference more prominent.

The poor performance of our method on the DrugBank dataset (Table 1) can be linked to the imbalance state of this dataset as shown in the Supplementary materials Section 4 (Fig.2). Our method has difficulty generalizing on DDI types with very low frequency. As future work, we aim to investigate this issue further in order to come up with an effective approach for handling DDI types with very low frequencies.

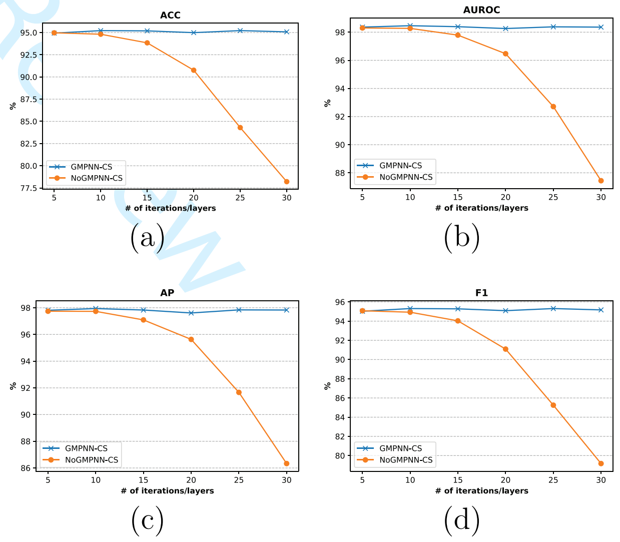


Fig. 5. Comparison of performance between our method GMPNN-CS, and its variant without edge gate values, NoGMPNN-CS, or equivalently with constant gate values all set to 1. Contrary to NoGMPNN-CS, GMPNN-CS can be extended to many iterations (layers) with constant performance.

6. Conclusion

We proposed GMPNN-CS, a computational method for drug-drug interaction (DDI) prediction. GMPNN-CS learns substructures of different sizes and shapes of drugs in order to infer whether a pair of drugs can cause a DDI based on their chemical substructures. We demonstrated empirically the effectiveness of GMPNN-CS using two real-world datasets, with significant performance improvement in one of them. Experiments are conducted in both transductive and inductive settings. A visual inspection (as conducted in the experiments) of substructure extraction and their involvement in a DDI prediction can be used as hint by both expert and non-expert users to interpret the results of a prediction.

7. Conflict of Interest

There is no conflict of interest.

8. Funding

This work was supported by National Nature Science Foundation of China (Grant No. 61872297) and Shaanxi Provincial Key Research & Development Program, China (Grand No. 2020KW-063).

9. Key points

- Using chemical substructures of drugs to predict drug-drug interaction between a pair of drugs by also giving the specific type of the interaction (multi-type prediction).
- Proposing a message passing neural network named gated message passing neural network (GMPNN) for learning substructures of various (or adaptive) sizes and shapes from molecular graphs of drugs.
- Using co-attention mechanism to learn the relevance of each pairwise substructure interaction involvement in a drug-drug interaction.
- Proposed method able to be applied both in transductive and inductive (or cold-start) settings.

References

1. Ryall KA and Tan AC. Systems biology approaches for advancing the discovery of effective drug combinations. *J. Cheminformatics*, 2015. 7(1).
2. Tatonetti NP, Ye PP, Daneshjou R et al. Data-driven prediction of drug effects and interactions. *Sci. Transl. Med.*, 2012. 4(125).
3. Sun X, Vilar S, and Tatonetti NP. High-throughput methods for combinatorial drug discovery. *Sci. Transl. Med.*, 2013. 5(205).
4. Harrold MW and Zavod RM. Basic Concepts in Medicinal Chemistry. *Drug Dev. Ind. Pharm*, 2014. 40(7):988–988.
5. Lu J, Yang J, Batra D et al. Hierarchical Question-Image Co-Attention for Visual Question Answering. In *NeurIPS*, volume 29. 2016 pages 289–297.
6. Zhang P, Wang F, Hu J et al. Label Propagation Prediction of Drug-Drug Interactions Based on Clinical Side Effects. *Scientific Reports*, 2015. 5(1):12339.
7. Yu H, Mao KT, Shi JY et al. Predicting and understanding comprehensive drug-drug interactions via semi-nonnegative matrix factorization. *BMC Syst. Biol.*, 2018. 12(S1):14.
8. Shi JY, Mao KT, Yu H et al. Detecting drug communities and predicting comprehensive drug–drug interactions via balance regularized semi-nonnegative matrix factorization. *J. Cheminformatics*, 2019. 11(1):28.
9. Masumshah R, Aghdam R, and Eslahchi C. A neural network-based method for polypharmacy side effects prediction. *BMC Bioinform.*, 2021. 22(1):1–7.
10. Deng Y, Xu X, Qiu Y et al. A multimodal deep learning framework for predicting drug–drug interaction events. *Bioinformatics*, 2020. 36(15):4316–4322.
11. Durant JL, Leland BA, Henry DR et al. Reoptimization of MDL keys for use in drug discovery. *J. Chem. Inf. Comput. Sci.*, 2002. 42(6):1273–1280.
12. Bolton EE, Wang Y, Thiessen PA et al. PubChem: Integrated Platform of Small Molecules and Biological Activities. *Annu. Rep. Comput. Chem.*, 2008. 4:217–241.
13. Gottlieb A, Stein GY, Oron Y et al. INDI: A computational framework for inferring drug interactions and their associated recommendations. *Mol. Syst. Biol.*, 2012. 8:592.
14. Zhang W, Chen Y, Liu F et al. Predicting potential drug-drug interactions by integrating chemical, biological, phenotypic and network data. *BMC Bioinform.*, 2017. 18(1):18.
15. Ferdousi R, Safdari R, and Omidi Y. Computational prediction of drug-drug interactions based on drugs functional similarities. *J. Biomed. Inform.*, 2017. 70:54–64.
16. Zhang W, Chen Y, Li D et al. Manifold regularized matrix factorization for drug-drug interaction prediction. *J. Biomed. Inform*, 2018. 88:90–97.
17. Vilar S, Harpaz R, Uriarte E et al. Drug-drug interaction through molecular structure similarity analysis. *J. Am. Med. Inform. Assoc.*, 2012. 19(6):1066–1074.
18. Ma T, Xiao C, Zhou J et al. Drug Similarity Integration Through Attentive Multi-view Graph Auto-Encoders. In *IJCAI*, volume 7. 2018 pages 3477–3483.
19. Veličković P, Casanova A, Liò P et al. Graph attention networks. In *ICLR*. 2018 .
20. Defferrard M, Bresson X, and Vandergheynst P. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *NeurIPS*, volume 29. 2016 pages 3844–3852.
21. Gilmer J, Schoenholz SS, Riley PF et al. Neural Message Passing for Quantum Chemistry. In *ICML*, volume 70 of *ICML'17*. 2017 pages 1263–1272.
22. Kipf TN and Welling M. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*. 2017 .
23. Duvenaud D, Maclaurin D, Aguilera-Iparraguirre J et al. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In *NeurIPS*, volume 28. 2015 pages 2224–2232.
24. Kearnes S, McCloskey K, Berndl M et al. Molecular graph convolutions: moving beyond fingerprints. *J. Comput.-Aided Mol. Des.*, 2016. 30(8):595–608.
25. Wu Z, Ramsundar B, Feinberg EN et al. MoleculeNet: A benchmark for molecular machine learning. *Chem. Sci.*, 2018. 9(2):513–530.
26. Yang K, Swanson K, Jin W et al. Analyzing Learned Molecular Representations for Property Prediction. *J. Chem. Inf. Model.*, 2019. 59(8):3370–3388.
27. Nyamabo AK, Yu H, and Shi JY. SSI–DDI: substructure–substructure interactions for drug–drug interaction prediction. *Brief. Bioinform.*, 2021. bbab133.
28. Xu N, Wang P, Chen L et al. MR-GNN: Multi-resolution and dual graph neural network for predicting structured entity interactions. In *IJCAI*, volume 2019-Augus. 2019 pages 3968–3974.
29. Huang K, Xiao C, Hoang TN et al. CASTER: Predicting Drug Interactions with Chemical Substructure Representation. *AAAI*, 2019. 34(01):702–709.
30. Deac A, Huang YH, Veličković P et al. Drug-Drug Adverse Effect Prediction with Graph Co-Attention. In *ICML Workshop on Computational Biology*. 2019 .
31. Ryu JY, Kim HU, and Lee SY. Deep learning improves prediction of drug–drug and drug–food interactions. *Proc. Natl. Acad. Sci. U. S. A.*, 2018. 115(18):E4304–E4311.
32. Wang H, Lian D, Zhang Y et al. Gognn: Graph of graphs neural network for predicting structured entity interactions. In *IJCAI*, volume 2. 2020 pages 1317–1323.
33. Feng YH, Zhang SW, and Shi JY. DPDDI: a deep predictor for drug-drug interactions. *BMC Bioinform*, 2020. 21(1):419.
34. Wu F, Souza A, Zhang T et al. Simplifying Graph Convolutional Networks. In *ICML*, volume 97. 2019 pages 6861–6871.
35. Wang Z, Zhang J, Feng J et al. Knowledge Graph Embedding by Translating on Hyperplanes. In *AAI*. 2014 pages 1112–1119.
36. Huang K, Fu T, Gao W et al. Therapeutics Data Commons: Machine Learning Datasets and Tasks for Therapeutics. *ArXiv*, 2021. abs/2102.0.
37. Zitnik M, Agrawal M, and Leskovec J. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*,

2018. 34(13):i457–i466.
38. Paszke A, Gross S, Massa F et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *NeurIPS*, volume 32. 2019 pages 8024–8035.
39. Fey M and Lenssen JE. Fast Graph Representation Learning with PyTorch Geometric. *ICLR Workshop*, 2019.
40. Kingma DP and Ba J. Adam: A Method for Stochastic Optimization. In *ICLR*. 2015 .
41. Hinton GE, Srivastava N, Krizhevsky A et al. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, 2012. abs/1207.0.
42. Huang SA and Lie JD. Phosphodiesterase-5 (PDE5) Inhibitors In the Management of Erectile Dysfunction. *Pharm. Ther.*, 2013. 38(7):407–419.

Arnold K. Nyamabo is currently pursuing his master's degree in computer science at Northwestern Polytechnical University. He is interested in graph representation learning and applications.

Hui Yu received the master's and Ph.D. degrees from Northwestern Polytechnical University, Xi'an, China, where he is currently an Associate Professor. His research interest includes bioinformatics, machine learning and data mining.

Zun Liu received the master's and Ph.D. degrees from Northwestern Polytechnical University, Xi'an, China, where he is currently a lecturer. His research interest includes bioinformatics, machine learning and information security.

JianYu Shi received his Ph.D. degree from Northwestern Polytechnical University, Xi'an, China, where he is currently a Professor. His research interests include bioinformatics, cheminformatics and artificial intelligence.

Supplementary Material

Drug-drug Interaction Prediction with Learnable Size-Adaptive Molecular Substructures

Arnold K. Nyambao Hui Yu Zun Liu JianYu Shi

1 Atom and Bond Features

The atom and bond features used in our model are given in Tables 1-2. They were all extracted using the cheminformatics python package RDKit[1].

Table 1: Atom Features. *: feature encoded in one-hot representation.

Feature	Size
Atom symbol (heavy atom type, e.g., C, O, N)	total # of heavy atoms in the dataset *
Atom degree (total # of directly-bonded neighbors)	10 *
Implicit valence	7 *
Formal charge	1
Number of radical electrons	1
Hybridaziation (sp, sp2, sp3, sp3d, or sp3d2)	5*
Aromacity	1

Table 2: Bond Features. All features are one-hot encoded.

Feature	Size
Bond type (single, double, triple, aromatic)	4
Conjugated	1
Is in ring	1

2 Implementation Details of non-linear functions in the proposed method

The implementation details of MLP$_{init-n}$, MLP$_{init\_e}$, and MLP$_w$ are given in Table 3. The implementation details of f_{sub}, MLP$_\gamma$, f_r, and f_{pred} are given in

[1]`https://www.rdkit.org/`

Table 4 . During training, we noticed that the outputs of residual connections of f_{sub}; that is, $\mathbf{x}^{(3)}$ and $\mathbf{x}^{(4)}$; tended to have very large values in magnitude, probability due to the summation. We therefore divided them by 2 to keep them as small as possible, in order also to avoid having larger gradient values (i.e., exploding gradients). FC: fully connected layer, BN: batch normalization layer, ρ: PReLU activation function.

Table 3: Implementation details of non-linear function modules in our proposed model. (Part 1)

$\mathrm{MLP}_{init-n}(\mathbf{x}^{(0)})$	$\mathrm{MLP}_{init\_e}(\mathbf{x}^{(0)})$	$\mathrm{MLP}_{w}(\mathbf{x}^{(0)})$
$\mathbf{x}^{(1)} = \mathrm{FC}(\mathbf{x}^{(0)})$ $\mathbf{x}^{(2)} = \mathrm{FC}(\rho(\mathbf{x}^{(1)}))$ $\mathbf{x}^{(3)} = \mathrm{FC}(\rho(\mathrm{BN}(\mathbf{x}^{(2)})))$ $\mathbf{x}^{(4)} = \mathrm{BN}(\mathbf{x}^{(3)})$	$\mathbf{x}^{(1)} = \mathrm{FC}(\mathbf{x}^{(0)})$	$\mathbf{x}^{(1)} = \mathrm{FC}(\mathbf{x}^{(0)})$ $\mathbf{x}^{(2)} = \mathrm{FC}(\rho(\mathbf{x}^{(1)}))$
$\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)} \in \mathbb{R}^{64}$	$\mathbf{x}^{(1)} \in \mathbb{R}^{64}$	$\mathbf{x}^{(0)}, \mathbf{x}^{(1)} \in \mathbb{R}^{64}$

Table 4: Implementation details of non-linear function modules in our proposed model. (Part 2)

$f_r / f_{pred}(\mathbf{x}^{(0)})$	$\mathrm{MLP}_{\gamma}(\mathbf{x}^{(0)})$	$f_{sub}(\mathbf{x}^{(0)})$
$\mathbf{x}^{(1)} = \mathrm{FC}(\rho(\mathbf{x}^{(0)}))$ $\mathbf{x}^{(2)} = \mathrm{FC}(\rho(\mathbf{x}^{(1)}))$	$\mathbf{x}^{(1)} = \mathrm{FC}(\mathbf{x}^{(0)})$ $\mathbf{x}^{(2)} = \mathrm{FC}(\rho(\mathbf{x}^{(1)}))$	$\mathbf{x}^{(1)} = \mathrm{FC}(\mathrm{BN}(\mathbf{x}^{(0)}))$ $\mathbf{x}^{(2)} = \mathrm{FC}(\rho(\mathrm{BN}(\mathbf{x}^{(1)})))$ $\mathbf{x}^{(3)} = (\mathrm{FC}(\rho(\mathrm{BN}(\mathbf{x}^{(2)}))) + \mathbf{x}^{(0)})\ /\ 2$ $\mathbf{x}^{(4)} = (\mathrm{FC}(\rho(\mathrm{BN}(\mathbf{x}^{(3)}))) + \mathbf{x}^{(3)})\ /\ 2$
$\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)} \in \mathbb{R}^{64}$	$\mathbf{x}^{(1)} \in \mathbb{R}^{128}$; $\mathbf{x}^{(2)} \in \mathbb{R}$	$\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)} \in \mathbb{R}^{128}$

3 Baseline methods: Implementation details

- **MR-GNN**: uses an LSTM (coined s-LSTM) to learn the most relevant resolutions of substructures of drugs, and another LSTM (i-LSTM) to jointly learn the most relevant combination of substructures from two drugs for DDI prediction. It is a multi-class classifier, contrary to our method which is a binary classification of DDI triplets (i.e, two drugs and an interaction type), it predicts the interaction type from a drug pair. For a fair comparison, we reimplemented it as binary classifier of DDI triplet. Furthermore, we observed better performance by removing the s-LSTM module.

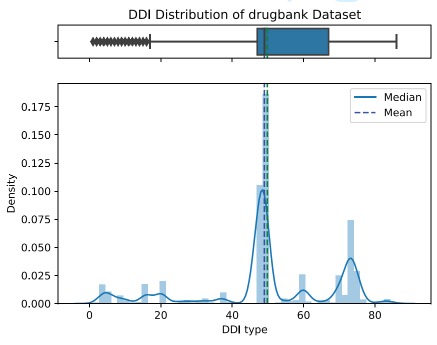
- **MHCADDI**: Uses co-attention mechanism to integrate joint drug-drug information during the representation learning of individual drugs. No significant modifications from the original work.

- **SSI-DDI**: considers each node hidden features as substructures and then computes interactions between these substructures to determine the final

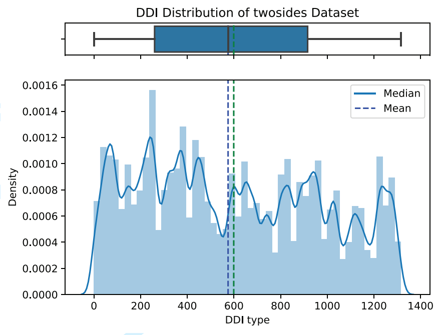
DDI prediction. For the Twosides dataset, we modify this method by incorporating information of the interaction type in the co-attention score computation as done with our model for better performance.

- **GAT-DDI**: baseline we implemented using a 5-layer graph attention network for drug representations which are obtained using global mean of the nodes as readout function. These representations are directly used for DDI prediction. For instance, for DDI triplet (d_x, r, d_y), let the final drug representations of d_x and d_y be respectively $\mathbf{g}_x$ and $\mathbf{g}_y$, and interaction r be represented as matrix $\mathbf{M}_r$, the prediction score is $P(d_x, r, d_y) = \sigma\left(\mathbf{g}_x^T \mathbf{M}_r \mathbf{g}_y\right)$.
- **GMPNN-U**: variant of our proposed method GMPNN-CS where the co-attention coefficient is simply uniform, each substructure pair has equal weight.

4 Drug-drug interaction type distribution



(a)



(b)

Figure 1: Distribution of DDI types of (a) DrugBank dataset and (b) Twosides dataset. The top boxes represent box plots drawn horizontally, and the bottom boxes are histograms.

Figure 1 shows the distribution of DDI types in the DrugBank (1(a)) and Twosides (1(b)) datasets. We can see that DrugBank is very imbalance (observed from the histogram plot) such that only a handful (observed from the box-plot) of DDI types constitute the vast majority of the dataset. Some DDI types barely occur 10 times (which explains first half of the box-plot is mainly composed of outliers) while others have a frequency of 60k+.

5 Performance per Drug-Drug Interaction Type

Figure 2 and Figure 3 show performance on each DDI type for each method on the DrugBank and Twosides datasets, respectively. For each metric, say, accuracy (ACC), the accuracy of a method (for instance, GMPNN-CS) on each DDI type is computed, and all these DDI type specific accuracies are represented collectively as a box-plot. We can see that on the DrugBank dataset (Figure 2) there are quite a few outliers (some with metric values as low as 0.5) for most of the methods, these are the metric values on the DDI types with very few frequencies, providing little data for them to be learned. On the Twosides dataset (See Figure 3, MHCADDI and GAT-DDI are not shown for the reasons given in the main manuscript), we can observe that our method, GMPNN-CS, outperforms all the other methods by a large margin, its lower outliers are even higher than metric values of some of the baseline methods. These figures show that our method performs relatively well on both datasets.

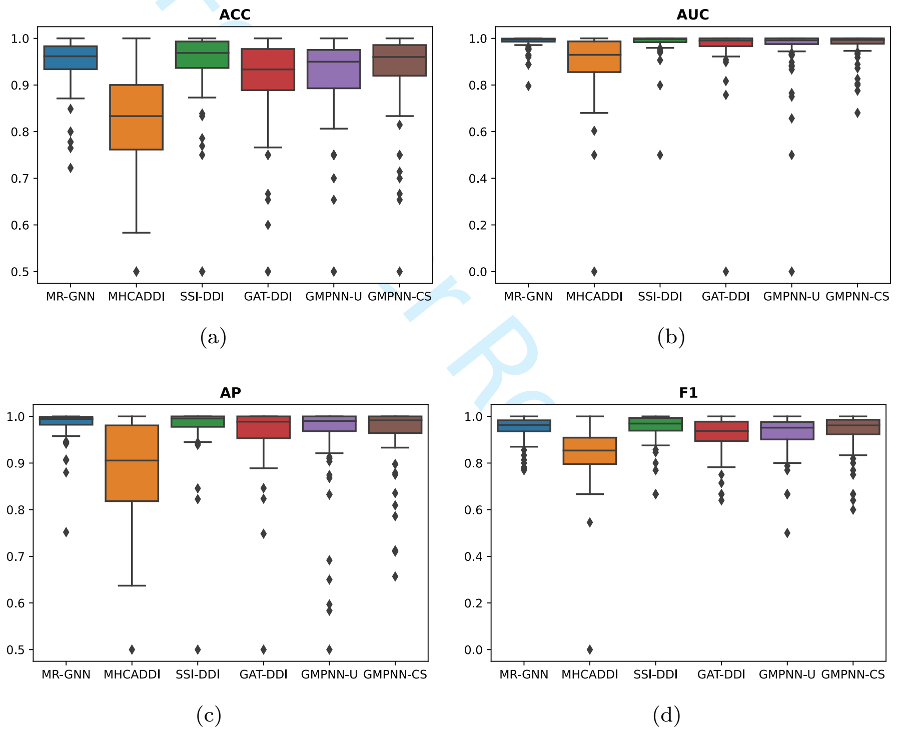


Figure 2: Performance of all the drug-drug interaction types on the DrugBank dataset for each method.

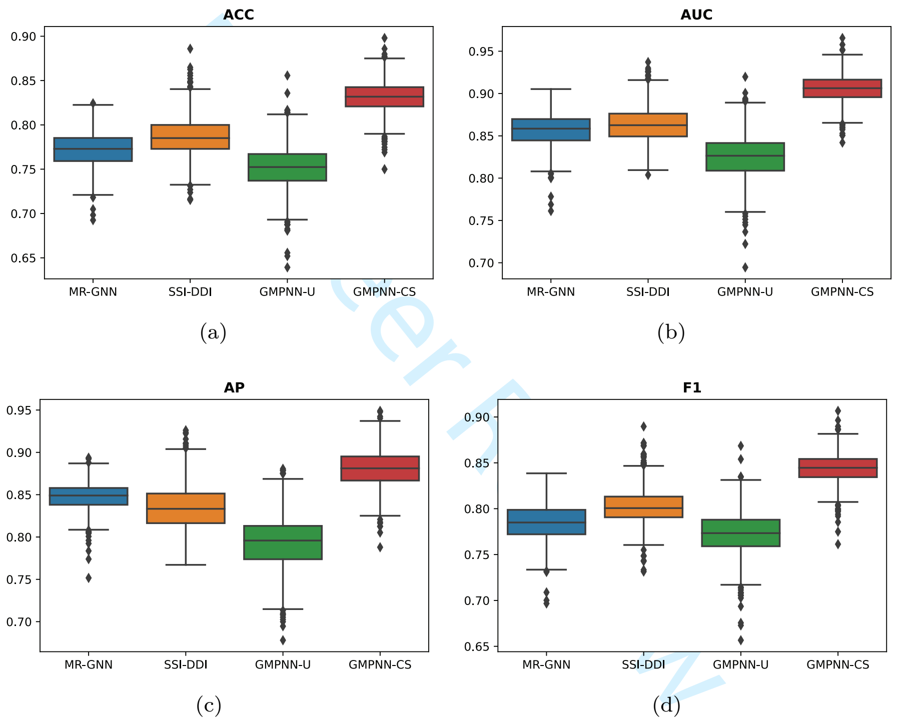


Figure 3: Performance of all the drug-drug interaction types on the Twosides dataset for each method.