SumGNN: Multi-typed Drug Interaction Prediction via Efficient Knowledge Graph Summarization

Yue Yu*[1], Kexin Huang*[2], Chao Zhang[1], Lucas M. Glass[3], Jimeng Sun[4], and Cao Xiao[3]

[1]College of Computing, Georgia Institute of Technology, Atlanta, GA
[2]Health Data Science, Harvard T.H. Chan School of Public Health, Boston, MA
[3]Analytic Center of Excellence, IQVIA, Cambridge, MA
[4]Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL

October 4, 2020

Abstract

Thanks to the increasing availability of drug-drug interactions (DDI) datasets and large biomedical knowledge graphs (KGs), accurate detection of adverse DDI using machine learning models becomes possible. However, it remains largely an open problem how to effectively utilize large and noisy biomedical KG for DDI detection. Due to its sheer size and amount of noise in KGs, it is often less beneficial to directly integrate KGs with other smaller but higher quality data (e.g., experimental data). Most of existing approaches ignore KGs altogether. Some tries to directly integrate KGs with other data via graph neural networks with limited success. Furthermore most previous works focus on binary DDI prediction whereas the multi-typed DDI pharmacological effect prediction is more meaningful but harder task.

To fill the gaps, we propose a new method SumGNN: *knowledge summarization graph neural network*, which is enabled by a subgraph extraction module that can efficiently anchor on relevant subgraphs from a KG, a self-attention based subgraph summarization scheme to generate reasoning path within the subgraph, and a multi-channel knowledge and data integration module that utilizes massive external biomedical knowledge for significantly improved multi-typed DDI predictions. SumGNN outperforms the best baseline by up to 5.54%, and performance gain is particularly significant in low data relation types. In addition, SumGNN provides interpretable prediction via the generated reasoning paths for each prediction.

*Equal Contribution

1 Introduction

Adverse drug-drug interactions (DDI) are modifications of the effect of a drug when administered with another drug, which is a common and dangerous scenario for patients with complicated conditions. Undetected adverse DDIs have become serious health threats and caused nearly 74, 000 emergency room visits and 195, 000 hospitalizations each year in the United States alone (Percha and Altman, 2013). To mitigate these risks and costs, accurate prediction of DDIs becomes a clinically important task. Two types of data are being utilized for developing DDI detection models: Manually curated DDI networks and large biomedical knowledge graphs.

Curated DDI networks: Researchers have curated DDI networks based on experimental datasets and literature such as TWOSIDES (Tatonetti et al., 2012), MINER (Zitnik et al., 2018b) and DrugBank (Wishart et al., 2008; Ryu et al., 2018). These curated data are *of higher quality but expensive to create and usually smaller in size.*

Knowledge Graph: Over the years, large knowledge graph (KG) such as (Rotmensch et al., 2017), Hetionet (Himmelstein and Baranzini, 2015) and DRKG (Ioannidis et al., 2020) have been constructed from literature mining and database integration. However, these KGs are *large and noisy*: out of their tens of thousands of nodes with millions of edges, only a small subgraph is relevant to a prediction target.

Deep Learning: Graph neural networks (GNN) have achieved great performance by casting DDI prediction as a link prediction problem on DDI graphs (Gysi et al., 2020; Zitnik et al., 2018a; Huang et al., 2020a; Lin et al., 2020). However, existing DL models are often trained only based on the DDI dataset at hand, ignoring the large biomedical knowledge graph (Ioannidis et al., 2020; Himmelstein and Baranzini, 2015) which can benefit the DDI predictions since DDI is driven by complicated biomedical mechanism. Some recent works (Karim et al., 2019; Lin et al., 2020) tried to integrate knowledge graph into the DDI prediction via direct integration of standard KG and GNN methods. But DDI prediction presents unique modeling difficulties since the input KG is large and noisy while the pertinent information for a drug pair is local. Moreover, most existing works also only make binary classification - predicting the presence of DDIs, despite that predicting the particular DDI type is a more meaningful task.

Our Approach. In this work, we propose a new method `SumGNN` that efficiently uses KG to aid drug interaction prediction. `SumGNN` enjoys improved predictive performance, efficiency, inductiveness and interpretability. `SumGNN` provides the following technical contributions:

1. **Local subgraph for identifying useful information**. We use local subgraph in the KG around drug pairs to extract useful information, instead of the entire KG. The subgraph formulation allows noise reduction by anchoring on relevant information and is highly scalable since the message passing receptive field is significantly decreased.
2. **Subgraph summarization scheme for generating reasoning path.** We then propose a summarization scheme to generate mechanism pathway for drug interactions. We develop a layer-independent self-attention mechanism to generate signal intensity score for each edge in the subgraph and create a KG pathway that has high scores. This pathway provides insights on the biological processes that drive drug interactions.
3. **Multi-channel data and knowledge integration for improved multi-typed DDI predictions**. We propose to use multi-channel neural encoding to aggregate diverse set of data sources, ranging from the summarized subgraph embedding to chemical structures. It enables utilization of massive external biomedical knowledge for significantly improved multi-typed DDI predictions. In addition, the neural encoding takes different subgraph in

each propagation, forming an inductive bias that promotes generalizability in low-resource DDI types.

We conduct extensive experiments to show SumGNN improves DDI prediction significantly. It has up to 5.54% increase over the best baseline while the inference time is greatly reduced. Moreover, SumGNN excels at low-resource settings whereas previous works do not. SumGNN is also able to provide reasonable clues about the underlying mechanism of the drug interactions.

2 Related Works

External knowledge graph integration. Recently, several efforts have attempted to leverage the KG for downstream tasks such as recommendation (Wang et al., 2019a,b; Gao et al., 2019), information extraction (Wang et al., 2018; Liang et al., 2020; Yu et al., 2020b) and drug interaction prediction (Celebi et al., 2019; Karim et al., 2019; Lin et al., 2020). (Celebi et al., 2019; Karim et al., 2019; Dai et al., 2020) project each entity and relation to a dense vector with knowledge graph embedding techniques (Bordes et al., 2013; Su et al., 2020; Trouillon et al., 2016) and then feed them to neural networks for prediction. However, they do not directly harness the neighborhood information for target entities during inference, thus the external knowledge information are not sufficiently exploited. (Lin et al., 2020) adopts graph convolutional networks with neighborhood sampling to explicitly model the neighborhood relations with higher inference speed. However, as each neighboring entity could play a crucial role in the drug interaction mechanism, random sampling could potentially dropout these important factors and hinders the prediction performance. In contrast, SumGNN provides a learnable way to extract useful information in the neighborhood. In addition, the previous works all focus on binary DDI prediction whereas SumGNN evaluates on multi-type relation network.

Subgraph Graph Neural Network. Graph neural networks have been proposed for modeling the relation between nodes (Kipf and Welling, 2017; Veličković et al., 2018; Schlichtkrull et al., 2018; Yu et al., 2020a; Srinivasa et al., 2020) and have been successfully applied to various domains (Shi et al., 2020; Yu et al., 2020c; Shang et al., 2019). Subgraph structure contains rich information for many graph learning tasks (Teru and Hamilton, 2020; Veličković et al., 2019; Huang and Zitnik, 2020). For instance, Ego-CNN applies local ego network to identify structures for graph classification (Tzeng and Wu, 2019). Alsentzer et al. (2020) formulate a multi-channel way to subgraph classification. Cluster-GCN (Chiang et al., 2019) and GraphSAINT (Zeng et al., 2020) use subgraphs to improve GNN scalability. More relevant to us, Zhang and Chen (2018) apply local subgraph for link prediction and Teru and Hamilton (2020) extend this idea into KG completion task via utilizing multi-relational information. In contrast, driven by the domain DDI prediction problem, SumGNN is the first to apply a graph summarization method on subgraphs to obtain tractable pathway and also with a new multi-channel neural encoding mechanism.

3 Method

We present SumGNN in this section. We summarize problem settings in Section 3.1 and describe our method in detail in Section 3.2. Our method can be decomposed into three modules. First, we use local subgraph in the KG around drug pairs to extract useful information. We then propose a summarization scheme to generate mechanism pathway for drug interactions. Then, we describe a multi-channel neural encoding layer to predict the pharmacological effect.

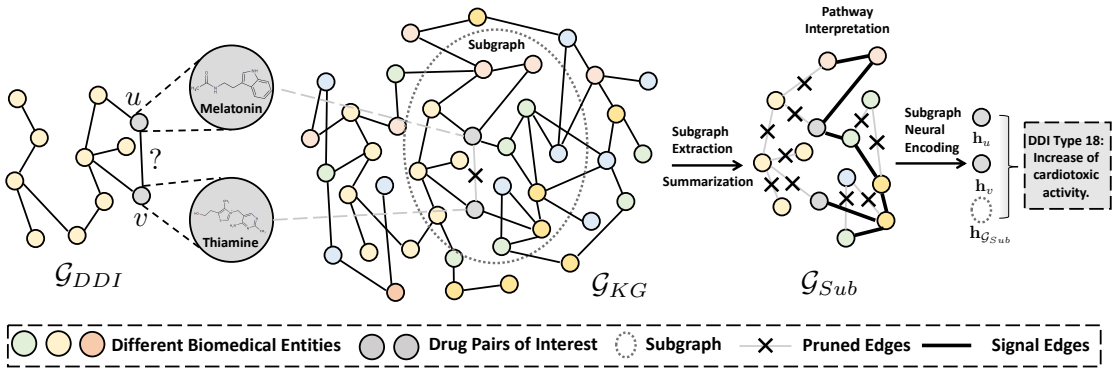


Figure 1: SumGNN illustration.

3.1 Problem Settings

Definition 1 (**Drug Interaction Graph**). *Given drugs $\mathcal{D}$ and pharmacological effects $\mathcal{R}_D$, the drug interaction graph $\mathcal{G}_{DDI}$ is defined as a set of triplets $\mathcal{G}_{DDI} = \{(u, r, v) \mid u \in \mathcal{D}, r \in \mathcal{R}_D, v \in \mathcal{D})\}$, where each triplet (u, r, v) represents that drug u and drug v have pharmacological effect r.*

Definition 2 (**External Biomedical Knowledge Graph**). *Given a set of various biomedical entities $\mathcal{E}$ and the biomedical relation among the entities $\mathcal{R}$, the external biomedical knowledge graph $\mathcal{G}_{KG}$ is defined as $\mathcal{G}_{KG} = \{(h, r, t) \mid h, t \in \mathcal{E}, r \in \mathcal{R}\}$ with each item (h, r, t) describes a biomedical relation r between entity h and entity t. Note that we aggregate the drug entities in $\mathcal{G}_{DDI}$ to $\mathcal{G}_{KG}$, i.e., $\mathcal{R}_D \in \mathcal{R}$, and $\mathcal{D} \in \mathcal{E}$.*

Problem 1 (**Multi-relational DDI Prediction**). *The Drug-drug interaction (DDI) prediction is to output the pharmacological effect given the a pair of drugs. Mathematically, it is to learn a mapping $\mathcal{F} : \mathcal{D} \times \mathcal{D} \rightarrow \mathcal{R}_D$ from a drug pair $(u, v) \in (\mathcal{D} \times \mathcal{D})$ to the pharmacological effect $r \in \mathcal{R}_D$.*

3.2 The SumGNN Method

SumGNN is composed of three modules: subgraph anchoring, knowledge summarization, and multi-channels neural encoding. For a given drug pairs, we anchor to a subgraph of potential biomedical entities that are close to the pairs in the KG. Then, we propose a new graph neural network that has a summarization scheme to provide a condense pathway to reason about drug interaction mechanism. Given this pathway graph, we use multi-channels neural encoding, to integrate diverse sources of available information to generate a sufficient drug pair representation. At last, a decoding classifier is followed to predict the interaction outcome. We initialize all entity embedding using KG method TransE (Bordes et al., 2013), where a entity is denoted as $\mathbf{h}_u^{(0)}$.

(A) The Local Subgraph Extraction Module

The biomedical KG describes the complicated mechanism of human biology. Modulation in several nodes (drug-pairs) in the KG can perturb the connected nodes (*e.g.* disease, cellular component, and etc.) which creates a ripple effect that eventually result in various physiological outcomes. The effect is diffused as distance between the drug pairs and the biomedical entities increases. Thus, to understand the drug interactions, we focus on local subgraphs in the KG

around the drug pairs. Specifically, for drug pairs u and v, we first extract the k-hop neighboring nodes for both u and v, $\mathcal{N}_k(u) = \{s \mid d(s,u) \leq k\}$ and $\mathcal{N}_k(v) = \{s \mid d(s,v) \leq k\}$, where $d(\cdot,\cdot)$ stands for the distance between two nodes on $\mathcal{G}_{KG}$. Then, we obtain the enclosing subgraph based on the intersection of these nodes, $\mathcal{G}_{Sub} = \{(u,r,v) \mid u,v \in \mathcal{N}_k(u) \cap \mathcal{N}_k(v), r \in \mathcal{R}\}$.

Motivated by (Zhang and Chen, 2018) which highlights the importance of node relative position to the central node u, v in the subgraph as capturing the rich structural information. Thus, we augment the initial node embedding in the subgraph by concatenating a position vector. For each node i, we compute the shortest path length $(d(i,u), d(i,v))$ between i and the center drug pairs nodes u, v. We convert it into a position vector $\mathbf{p}_i = [\text{one-hot}(d(i,u)) \oplus \text{one-hot}(d(i,v))]$. Then, we update the node i representation as $\mathbf{h}_i^{(0)} = [\mathbf{h}_i^{(0)}, \mathbf{p}_i]$.

(B) The Knowledge Summarization Module

To provide biological insights in addition to the predictive outcome, we design a knowledge summarization module to summarize the subgraph information into a pathway for potential drug interactions. This summarization means that we need to retrieve paths that contain signals for the interactions while remove paths that are not important. To achieve this, we adopt a layer-independent relation-aware self-attention to assign a weight for every edges in $\mathcal{G}_{Sub}$. These weights are generated based on the input featurization $\mathbf{h}^{(0)}$ and represent the interaction signal intensities for edge pruning.

We denote the interaction signal intensity score $\alpha_{i,j}$ as the edge connecting any biomedical entity i and j. Inspired by the relation-aware transformer architecture (Shaw et al., 2018), we use self-attention mechanism, which takes account into all nodes in the subgraph to generate the attention weight. This attention mechanism is ideal for us because it means that the signal intensity score is generated after examining all biomedical entities in the subgraph around the drug-pairs. Here, $\alpha_{u,v}$ is calculated as

$$\alpha_{i,j} = \text{Threshold}\left(\sigma\left(\frac{\mathbf{h}_j^{(0)}\mathbf{W}^J(\mathbf{h}_i^{(0)}\mathbf{W}^I + \mathbf{r}_{ij})^T}{\sqrt{d_k}}\right), \gamma\right), \tag{1}$$

where the $\mathbf{W}^J$ and $\mathbf{W}^I$ are the self-attention key weights that contain representation for each node in the subgraph, r_{ij} encodes the relationship between the two nodes i and j, $\sqrt{d_k}$ is the size of feature vector $\mathbf{h}^{(0)}$ for normalization, γ is the signal threshold and $\sigma(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ is the *tanh* function for non-linear transformation.

Intuitively, this function first computes the dot product between $\mathbf{h}_i^{(0)}$ and $\mathbf{W}^I$ to get attention score between node i and every other node in the subgraph. Then we sum it up with the relation embedding, followed up the same procedure to calculate attention score between node j and every other node through dot product between $\mathbf{h}_j^{(0)}$ and $\mathbf{W}^J$. By taking the dot product with every other nodes for both i, j, the final score leverages considers all the subgraph information. Then, after non-linear transformation, we obtain the signal intensity score ranging from -1 to 1 for this edge. At last, we apply a threshold function to screen out edges that are below an intensity score threshold γ by setting them with weight 0 since they are not important for the interaction prediction and setting them 0 would prune these edges from message passing in the graph neural network. This step is applied to every edge in the subgraph.

Note that existing graph attention approaches (Veličković et al., 2018; Cai and Lam, 2020; Shaw et al., 2018) generate attention weights for every edge in every layer. However, this way can provide potentially contradicting signals across layers for the same edge, precluding the

generation of interpretable pathways. In the contrast, SumGNN adopts a layer-independent attention mechanism, which only depends on the fixed input embedding to prune edges first. It applies an unequivocal pathway for model explainability.

(C) The Multi-Channel Integration Module

To obtain a powerful representation for drug interaction prediction, we integrate a diverse set of information sources.

Channel 1: Summarized Knowledge Using the knowledge summarization, we identify a summarized subgraph that is important to input drug pairs. We want to generate latent representation that leverage this subgraph for the input drug pairs. We integrate it using the following message passing scheme. For a node v, we compute a relation-aware message weighted by the signal intensity score $\mathbf{b}_v^l$ at layer l using

$$\mathbf{b}_v^{(l)} = \sum_{u \in \mathcal{N}_v} \alpha_{u,v}^{(l)} \left(\mathbf{h}_u^{(l-1)} \mathbf{W}_r^{(l)} \right), \tag{2}$$

where $\mathcal{N}_v$ denotes the neighbors of node v in subgraph $\mathcal{G}_{Sub}$, $\mathbf{W}_r^{(l)}$ is the weight matrix to transform hidden representation for node u, v's relation r in layer l. To avoid overfitting, we use basis decomposition (Schlichtkrull et al., 2018) to decompose $\mathbf{W}_r^{(l)}$ into the linear combination of a small number of basis matrices $\{\mathbf{V}_b\}_{b \in B}$ as

$$\mathbf{W}_r^{(l)} = \sum_{b=1}^{B} a_{rb}^{(l)} \mathbf{V}_b^{(l)}. \tag{3}$$

Then, we propagate the message $\mathbf{b}_v^{(l)}$ to the updated representation $\mathbf{h}_v^{(l)}$ of node v via

$$\mathbf{h}_v^{(l)} = \text{ReLU} \left(\mathbf{W}_{\text{self}}^{(l)} \mathbf{h}_v^{k-1} + \mathbf{b}_v^{(l)} \right), \tag{4}$$

where $\mathbf{W}_{\text{self}}$ is the weight matrix to transform the node embedding itself.

Channel 2: Subgraph Features To obtain the embeddings for subgraphs (denoted as $\mathbf{h}_{\mathcal{G}_{Sub}}$), we take the average of all node embeddings in $\mathcal{G}_{Sub}$ at layer l projected by a linear layer as

$$\mathbf{h}_{\mathcal{G}_{Sub}}^{(l)} = \text{Mean} \left(\mathbf{W}_{\text{Sub}} \mathbf{h}_i^{(l)} \right). \tag{5}$$

Channel 3: Drug Fingerprint Molecular information such as chemical fingerprints have shown to be powerful predictor of drug interactions (Huang et al., 2020b). Thus, in addition to the network representation, we obtain the Morgan fingerprint $\mathbf{f_v}$ (Rogers and Hahn, 2010), which is a predictive descriptor of drugs, for each drug v.

Layer-wise Channels Aggregation To assemble various representation generated via each layer, we adopt the layer-aggregation mechanism (Xu et al., 2018). We concatenate node/subgraph embeddings in every layer, *i.e.*, $\mathbf{h}_v = [\mathbf{h}_v^{(1)}, \mathbf{h}_v^{(2)}, \cdots, \mathbf{h}_v^{(L)}]$ and $\mathbf{h}_{\mathcal{G}_{Sub}} = [\mathbf{h}_{\mathcal{G}_{Sub}}^{(1)}, \mathbf{h}_{\mathcal{G}_{Sub}}^{(2)}, \cdots, \mathbf{h}_{\mathcal{G}_{Sub}}^{(L)}]$ where is L is the layer size. To integrate chemical fingerprints, we update the layer-aggregated embedding by concatenation of chemical representation: $\mathbf{h}_v = [\mathbf{h}_v \oplus \mathbf{f}_v]$.

At last, we combine the various channels together to obtain the input drug-pairs representation $\mathbf{h}_{u,v} = [\mathbf{h}_u, \mathbf{h}_v, \mathbf{h}_{\mathcal{G}_{Sub}}]$. To predict the relation, we obtain a prediction probability vector $\mathbf{p}_{u,v}$

where each value in the vector corresponds to a the likelihood of a relation. $\mathbf{p}_{u,v}$ is computed via feeding the drug pair representation to a decoder parameterized by $\mathbf{W}_{\text{pred}}$:

$$\mathbf{p}_{u,v} = \mathbf{W}_{\text{pred}}\mathbf{h}_{u,v}. \tag{6}$$

3.2.1 Training and Inference

During training, for multi-class classification task, we adopt the the cross entropy loss ℓ_{CE} for each edge (u, r, v) as

$$\ell_{\text{CE}}(u, r, v) = -\sum_{r=1}^{R} \log(\hat{y}_r) \cdot y_r \tag{7}$$

where $\hat{y}_r = \text{softmax}(\mathbf{p}^r_{u,v}) = \frac{\exp(\mathbf{p}^r_{u,v})}{\sum_{i=1}^{R}\exp(\mathbf{p}^i_{u,v})}$ and y_r is the binary indicator if class r is the correct label for u and v. For multi-label classification task, given the edge (u, r, v), we adopt the binary cross entropy loss ℓ_{BCE} as

$$\ell_{\text{BCE}}(u, r, v) = -\log \hat{y}_r - \mathbb{E}_{w\sim P_w(v)} \log\left(1 - y^{u,w}_r\right) \tag{8}$$

where (u, r, w) is the sampled negative edge for relation r. This is achieved by replacing node v to node w that is sampled randomly according to a distribution $P_w(v) \propto d_w(v)^{3/4}$ (Mikolov et al., 2013). Then $\hat{y}_r = \text{sigmoid}(\mathbf{p}^r_{u,v})$, $y^{uw}_r = \text{sigmoid}(\mathbf{p}^r_{u,w})$ is the prediction score for two edges. Considering all edges, the final loss $\mathscr{L}$ in `SumGNN` is

$$\mathscr{L} = \sum_{(u,r,v)\in\mathscr{E}} \ell(u, r, v) \tag{9}$$

where ℓ is either Eq. (7) or Eq. (8) depending on the task type. During training, we learn the model parameter by minimizing the total loss $\mathscr{L}$ using stochastic gradient optimizers such as Adam (Kingma and Ba, 2014).

During inference, an unseen node pair u, v's subgraph in the KG is extracted and fed into the same pipeline to calculate the relation vector. For multi-class task, we use the highest probability relation as the predicted relation and for multi-label task, we collect all scores from both positive and negative counterparts for all relations.

4 Experiments

4.1 Experiment Setup

Datasets (1) DrugBank dataset (Wishart et al., 2008) contains 1,709 drugs (nodes) and 136,351 drug pairs (edges), which are associated with 86 types of pharmacological relations between drugs, such as increase of cardiotoxic activity, decrease of serum concentration and etc. Each drug pair can contain one or two relations. As more than 99.8% of edges have only one edge type (Ryu et al., 2018), we filtered the edge with more than one type in our study. (2) TWOSIDES (Tatonetti et al., 2012) dataset contains 645 drugs (nodes) and 46,221 drug-drug pairs (edges) with 200 different drug side effect types as labels. For each edge, it may be associated with multiple labels. Following (Zitnik et al., 2018a; Dai et al., 2020), we keep 200 *commonly-occurring* DDI types ranging from Top-600 to Top-800 to ensure every DDI type has at least 900 drug combinations. (3) For external knowledge base, we use Hetionet (Himmelstein and Baranzini,

2015), which is a large heterogeneous knowledge graph merged from 29 public databases. To **ensure no information leakage**, we remove all the overlapping DDI edges between Hetionet and the dataset. In the end, we obtain 33,765 nodes out of 11 types (e.g., gene, disease, pathway, molecular function and etc.) with 1,690,693 edges from 23 relation types.

Baselines We compare our models with several baselines[1].

- MLP (Rogers and Hahn, 2010) uses a two-layer MLP on Morgan fingerprint to directly predict drug interactions.
- Node2vec (Grover and Leskovec, 2016) first learns the embeddings for drugs in the network. Then, it predict the relation for drug pairs via a linear layer over embeddings.
- Decagon (Zitnik et al., 2018a) adopts multi-relational graph convolutional network (Schlichtkrull et al., 2018) on the DDI network for drug interaction prediction.
- GAT (Veličković et al., 2018) uses attention networks to aggregate neighborhood information in DDI network.
- SkipGNN (Huang et al., 2020a) predicts drug interactions by aggregating information from both direct interactions and second-order interactions via two GNNs.
- KG-DDI (Karim et al., 2019) first extracts KG embeddings for drugs in the KG, then adopts a Conv-LSTM model using the embeddings for drug interaction prediction.
- GraIL (Teru and Hamilton, 2020) is for inductive relation prediction on KGs, which uses local subgraph.
- KGNN (Lin et al., 2020) samples and aggregates neighborhoods for each node from their local receptives via GNN and with external knowledge graph.

Metrics The task on the DrugBank dataset is a multi-class classification, thus we consider the following metrics:

- Macro F1: average F1 score over *different classes* as Macro F1 $= \frac{1}{N}\sum_{k=1}^{N}\frac{2P_k \cdot R_k}{P_k+R_k}$, where N is the # of classes and P_k, R_k is the precision and recall for k-th class.
- Micro F1: weighted average F1 score where the weight of a class is the fraction of samples of the class out of the entire samples, i.e. Micro F1 $= \sum_{k=1}^{N}\frac{\mathcal{N}_k}{\mathcal{N}}\frac{2P_k \cdot R_k}{P_k+R_k}$, where $\mathcal{N}_k, P_k, R_k$ is the number of samples, precision and recall of class k and $\mathcal{N}$ is the total size of the dataset.
- Cohen's Kappa (Cohen, 1960) measures the inter-annotator agreement as $\kappa = (p_o - p_e)/(1 - p_e)$, where p_o is the observed agreement (identical to accuracy), p_e is the probability of randomly seeing each class.

The task on the TWOSIDES dataset is a multi-label prediction. We follow (Zitnik et al., 2018a) and consider the following measure. For each side effect type, we calculate the performance individually and use the average performance over all side effects as the final result.

- ROC-AUC is the average area under the receiver operating characteristics curve as ROC-AUC $= \sum_{k=1}^{n} \mathrm{TP}_k \,\Delta\, \mathrm{FP}_k$, where k is k-th true-positive and false-positive operating point $(\mathrm{TP}_k, \mathrm{FP}_k)$.
- PR-AUC is the average area under precision-recall curve PR-AUC $= \sum_{k=1}^{n} \mathrm{Prec}_k \,\Delta\, \mathrm{Rec}_k$ where k is k-th precision/recall operating point $(\mathrm{Prec}_k, \mathrm{Rec}_k)$.
- AP@50 is the average precision at 50, where AP@$k = \frac{|Y_k \cap \hat{Y}_k|}{|Y_k|}$, Y_k is the predicted labels at k and $\hat{Y}_k$ are the ground-truth labels.

[1]Further details on baseline methods, implementation and parameters are in the supplementary.

Table 1: SumGNN achieves the best predictive performance compared to state-of-the-art baselines in DDI prediction. Average and standard deviation of five runs are reported. For these metrics, higher values always indicate better performance.

Dataset	**Dataset 1: DrugBank**			**Dataset 2: TWOSIDES**		
Classification Task	Multi-class			Multi-label		
Methods	Macro F1	Micro F1	Cohen's Kappa	ROC-AUC	PR-AUC	AP@50
MLP (Rogers and Hahn, 2010)	61.10±0.38	82.14±0.33	80.50±0.18	82.60±0.26	81.23±0.14	73.45±0.28
Node2Vec (Grover and Leskovec, 2016)	24.92±0.32	71.09±0.40	63.79±0.37	90.66±0.13	88.87±0.23	83.00±0.30
Decagon (Zitnik et al., 2018a)	57.35±0.26	87.19±0.28	86.07±0.08	91.72±0.04	90.60±0.12	82.06±0.45
GAT (Veličković et al., 2018)	33.49±0.36	77.18±0.15	74.20±0.23	91.18±0.14	89.86±0.05	82.80±0.17
SkipGNN (Huang et al., 2020a)	59.66±0.26	85.83±0.18	84.20±0.16	92.04±0.08	90.90±0.10	84.25±0.25
KG-DDI (Karim et al., 2019)	36.39±0.32	82.48±0.12	78.89±0.27	90.75±0.07	88.16±0.12	83.48±0.05
GraIL (Teru and Hamilton, 2020)	81.31±0.30	89.89±0.24	88.07±0.20	92.89±0.09	91.10±0.19	86.21±0.05
KGNN (Lin et al., 2020)	73.99±0.11	90.89±0.20	89.64±0.24	92.84±0.07	90.78±0.20	86.05±0.12
SumGNN (Ours)	**86.85±0.44**	**92.66±0.14**	**90.72±0.13**	**94.86±0.21**	**93.35±0.14**	**88.75±0.22**
SumGNN-KG	78.35±0.51	89.05±0.36	87.28±0.08	92.62±0.13	90.80±0.40	85.75±0.10
SumGNN-Sum (w/o Summarization)	83.20±0.34	90.83±0.19	90.14±0.10	94.09±0.16	92.55±0.24	87.65±0.24
SumGNN-SF (w/o Subgraph Features)	84.47±0.22	91.88±0.21	90.26±0.19	93.94±0.11	92.45±0.22	87.69±0.08
SumGNN-CF (w/o Chemical Features)	83.57±0.36	91.31±0.17	90.07±0.11	94.35±0.11	92.86±0.20	88.10±0.07

Evaluation Strategy. For both datasets, we split it into 7:1:2 as train, development and test set. For the DrugBank dataset, since the label distribution is highly imbalanced, we ensure train/dev/test set contain samples from all classes. For the TWOSIDES dataset, we use the same method in (Zitnik et al., 2018a) to generate negative counterparts for each positive edge by sampling the complement set of positive examples. For every experiment, we conduct five independent runs and select the best performing model based on the loss value on the validation set.

4.2 Main Results: SumGNN achieves superior performance

We report the performance of our model and all baselines in table 1. From the result, we find that SumGNN achieves the best performance in DDI prediction on two datasets, accurately predicting the correct DDI pharmacological effect consistently. Particularly, SumGNN has 27.19%, 5.47%, 4.65% absolute increase over the best baseline without KG on three metrics respectively on DrugBank dataset and 2.84%, 2.45%, 4.50% increase on TWOSIDES dataset. Also, SumGNN achieves 5.54%, 1.77%, 1.08% and 1.97%, 2.25%. 2.54% absolute increase over the state-of-the-art baselines with KG on two datasets. These results clearly verifies the efficacy of our method.

By comparing the latter four methods with the former five methods, we find the use of KG significantly improves DDI performance, highlighting the necessity of external knowledge usage. Comparing SumGNN with KGNN and KG-DDI, we show that simply adopting KG embeddings as well as neighborhood sampling are *insufficient to fully harness the KG information*, whereas SumGNN provides the best approach to leverage the external KG and also corroborates with our motivation that use of subgraph reduces noise and irrelevant information. Although GraIL also extracts subgraphs for downstream tasks, they merely consider the position information while neglecting the multi-channel features during information propagation. Therefore, their performance is still limited for drug interaction prediction. In addition, GraIL does not use

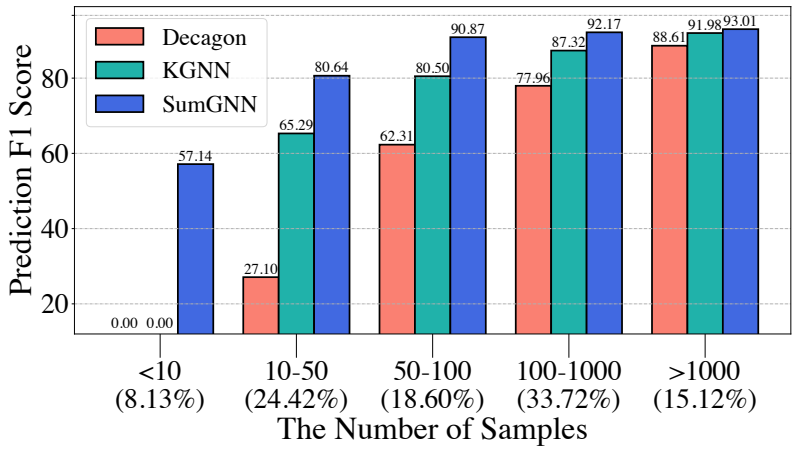


Figure 2: The average F1 score for different relations with various number of samples in the training set. Here, the x-axis indicate the number of samples as well as the ratio of classes in this group to all classes.

any knowledge summarization techniques, which potentially further eliminating the irrelevant information in the local subgraph.

Another interesting finding is that the improvement of SumGNN is clearly *more significant* on DrugBank than TWOSIDES dataset. We take a closer look at this problem and find that the major difference of these two dataset is that the DrugBank is more imbalanced – more than 30% of the relation types occurs less than 50 times in the training set while more than 10% of the relation types occurs more than 1000 times. To examine the model's prediction performance on the size of training data associated with each relation, we first split the relations types into 5 groups with different number of training data and then plot the average F1 score of these group in Figure 2. By comparing the performance of SumGNN against the strongest baseline (KGNN) and the variant of no KG. (Decagon), we find that using KG can effectively boost the performance when the samples are scarce, as injecting KG can bring at least 38.19% improvement on F1 score for relations occuring less than 50 times but less than 5% for relations more than 1000 times. In addition, compared with neighborhood sampling approaches proposed in KGNN, SumGNN can achieve *significantly better* performance when the samples are *extremely scarce* (e.g. few-shot settings). When the training samples are less than 10, both decagon and KGNN cannot give correct predictions, while our model can still achieve 57.14% accuracy. One potential reason is that SumGNN feeds different subgraphs in every GNN propagation, which forms a much-needed inductive bias over unseen subgraphs. This is in sharp contrast to previous approaches such as KGNN. It also justifies that SumGNN's knowledge summarization via subgraphs is more effective to harness the external knowledge.

4.3 Ablation Study

To study the usage of KG, we remove the knowledge graph (SumGNN-KG) and perform prediction on the DDI graph. We see SumGNN has 8.5% absolute increase in Macro F1 on DrugBank, highlighting the usefulness of KG. To evaluate the knowledge summarization module, we remove the summarization component (SumGNN-Sum) and use the raw local subgraph to predict the outcome. We see SumGNN has 3.65% increase for DrugBank and 2.24% increase for TWOSIDES on Macro F1, suggesting that the summarization further condenses the relevant information and elevate the performance.

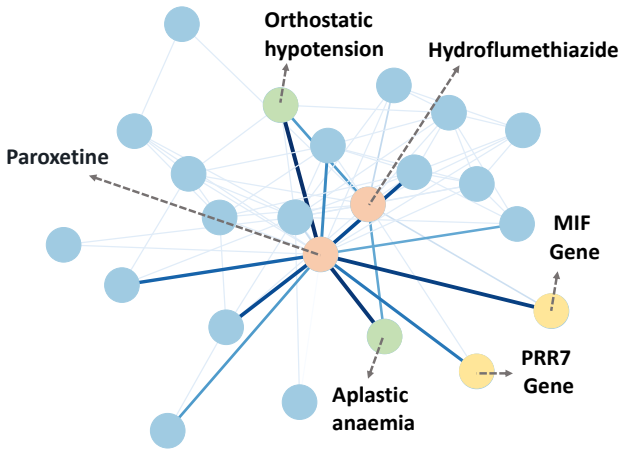


Figure 3: SumGNN generates a short reasoning path to provide clues for understanding drug interactions. Low-weight edges in the extracted subgraph are filtered out by SumGNN and SumGNN focus on a sparse set of signal edges and nodes.

To study the effect of multi-channel neural encoding, we compare the result of SumGNN with several variants that remove specific channel information (i.e. subgraph features and chemical features), and we find that these channel information all contribute to the overall performance. Particularly, after removing the subgraph embedding (SumGNN-SF), the Macro F1 drops by 2.38% on DrugBank and 0.92% on TWOSIDES respectively. When removing chemical fingerprint (SumGNN-CF), the performance also degrades 3.28% on Macro F1 for DrugBank, corroborating with the indispensable roles of each of the channel.

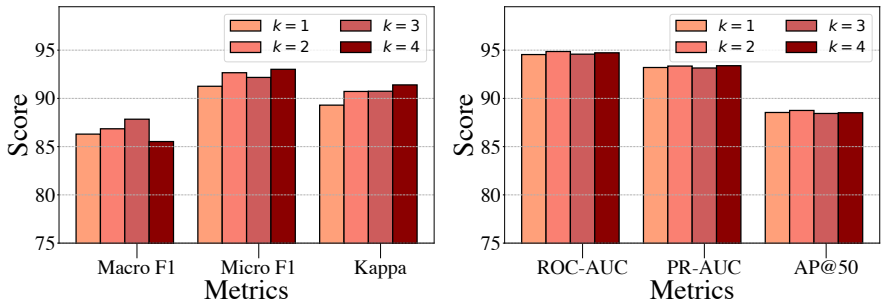
4.4 Case Study

Using the external knowledge summarization module, we are able to discover signal edges, which provide biological pathways to hint at the potential mechanism of DDIs. We provide a case study of a drug pair in the test set, Paroxetine and Hydroflumethiazide. SumGNN accurately predicts the correct DDI type "increase of the central nervous system depressant activities". We then visualize the generated pathway from SumGNN's summarization scheme in Fig 3. We see that the model significantly reduces irrelevant nodes and edges in the subgraph of the KG and focuses on a sparse set of nodes to make prediction. We examine the nodes that have high signals connection to the target pairs and find literature evidence support. Particularly, both drug pairs connect strongly to two side effects nodes, orthostatic hypotension and aplastic anaemia. Notably, orthostatic hypotension is closely related to multiple system atrophy, a central nervous system problem (Jones et al., 2015). As both drugs incur risk in the side effects, they could be aggravated when these drugs are taken together. This case study illustrates how to use SumGNN for potential DDI prediction.

4.5 Parameter studies

We study the effect of key parameters. When evaluating one parameter, we fix other parameters to their default values.

• **Effect of the hop of the subgraph** k: Figure 4 shows the result of SumGNN with varying k. From the result, we find that for DrugBank dataset, the performance first increase when k is



(a) Result of DrugBank (b) Result of TWOSIDES

Figure 4: Model performance given different parameter k.

Table 2: The running time for one epoch of SumGNN for two datasets. Note that **w/o Subgraph** is a variant that directly aggregates the information for all neighbors on KG.

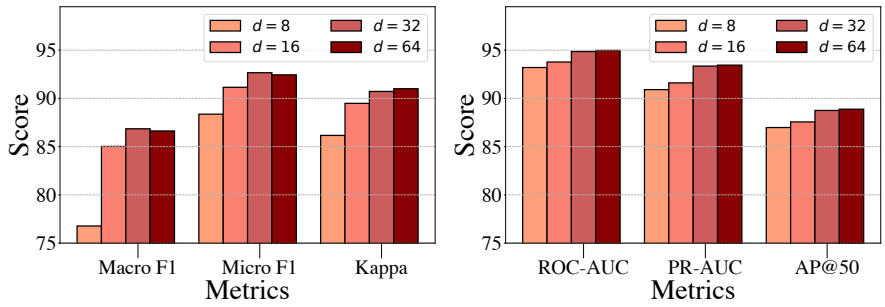
Dataset	**Size of Hop** k				**w/o Subgraph**
	1	2	3	4	
DrugBank	132 s	141 s	178 s	205 s	1279 s
TWOSIDES	62 s	75 s	91 s	102 s	471 s

small. However, when k increases from 3 to 4, we observe slight performance drops on Macro F1 score. These indicate that the larger subgraph can bring more useful information while when k is too large, it may also bring some noise and hurt the performance. For TWOSIDES dataset, we find the result is more stable with different k, indicating even 1-hop subgraph provides adequate information for DDI task.

Moreover, to show how **subgraph formulation drives efficiency**, we compare the training time between SumGNN with varying subgraph size and SumGNN with the entire KG to propagate (See Table 2). We find SumGNN saves 80% of training time via subgraph anchoring, which demonstrates the efficiency of our approach.

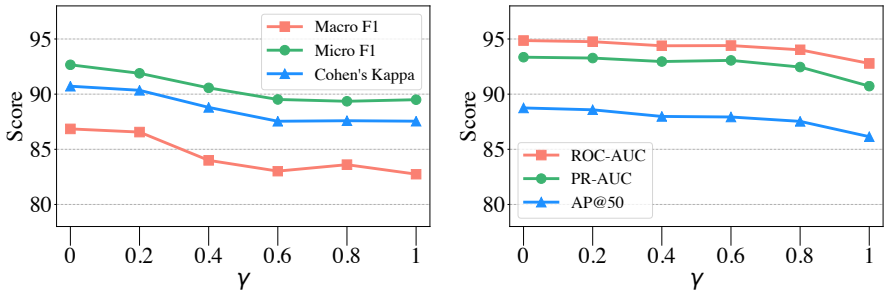
• **Effect of the dimension of embeddings** d: Figure 5 exhibits the influence of embedding dimension d. The result indicates that when d is small, increasing d can boost the performance. But when d becomes large, the gain is marginal.

• **Effect of the threshold for summarization** γ: Figure 6 shows the result of SumGNN with different threshold γ, which demonstrates that when γ is small, the performance is rather stable as filtering edges with low score has little effect on the final prediction. This also means SumGNN is able to achieve similar predictive performance while removing many irrelevant edges. However, when γ is large ($\gamma > 0.4$ for DrugBank and $\gamma > 0.8$ for TWOSIDES), it is clear that the performance drops more. In such cases, the summarized graph is more sparse and we might filter out potential useful edges. To sum up, there is a trade-off between the explainbility and performance.



(a) Result of DrugBank (b) Result of TWOSIDES

Figure 5: Model performance given different parameter d.



(a) Result of DrugBank (b) Result of TWOSIDES

Figure 6: Model performance given different parameter γ.

5 Conclusion

In this paper, we propose a new method `SumGNN`: *knowledge summarization graph neural network* for multi-typed DDI predictions, which is mainly enabled by a local subgraph module that can efficiently anchor on relevant subgraphs from a KG, a self-attention based subgraph summarization scheme that can generate reasoning path within the subgraph, and a multi-channel knowledge and data integration module that utilizes massive external biomedical knowledge for significantly improved multi-typed DDI predictions. Experiments on real-world datasets demonstrated the strong performance of `SumGNN`.

6 Ethics Statement

Adverse drug interactions incur morbidity and mortality for patients and are also one of the leading cause for failure of clinical trials, preventing promising life-saving drugs to go to the market. To test on the combination of all approved drugs in wet-lab setting is unfeasible, thus, in-silico approaches provide an alternative way to accurately capturing the interactions.

In addition, computational approaches depend heavily on the training data. If the number of training data associated with one specific drug interaction type is low, it is difficult to predict accurately. In contrast to other works, `SumGNN` is able to generate good performance in low-resource settings. `SumGNN` is also a general framework and can be adapted to predict any other interactions such as drug-disease interaction. The ability to low-resource learning could also

mean to excel at finding drugs for rare diseases.

References

Alsentzer, E., Finlayson, S. G., Li, M. M., and Zitnik, M. (2020). Subgraph neural networks. *NeurIPS*.

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., and Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In *NeurIPS*.

Cai, D. and Lam, W. (2020). Graph transformer for graph-to-sequence learning. In *AAAI*.

Celebi, R., Uyar, H., Yasar, E., Gumus, O., Dikenelli, O., and Dumontier, M. (2019). Evaluation of knowledge graph embedding approaches for drug-drug interaction prediction in realistic settings. *BMC Bioinformatics*.

Chiang, W.-L., Liu, X., Si, S., Li, Y., Bengio, S., and Hsieh, C.-J. (2019). Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *KDD*.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*.

Dai, Y., Guo, C., Guo, W., and Eickhoff, C. (2020). Wasserstein adversarial autoencoders for knowledge graph embedding based drug-drug interaction prediction. *arXiv:2004.07341*.

Gao, C., Huang, C., Yu, Y., Wang, H., Li, Y., and Jin, D. (2019). Privacy-preserving cross-domain location recommendation. *IMWUT*.

Grover, A. and Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In *KDD*.

Gysi, D. M., Valle, Í. D., Zitnik, M., Ameli, A., Gan, X., Varol, O., Sanchez, H., Baron, R. M., Ghiassian, D., Loscalzo, J., et al. (2020). Network medicine framework for identifying drug repurposing opportunities for covid-19. *arXiv:2004.07229*.

Himmelstein, D. S. and Baranzini, S. E. (2015). Heterogeneous network edge prediction: a data integration approach to prioritize disease-associated genes. *PLoS Computional Biology*.

Huang, K., Xiao, C., Glass, L., Zitnik, M., and Sun, J. (2020a). Skipgnn: Predicting molecular interactions with skip-graph networks. *arXiv:2004.14949*.

Huang, K., Xiao, C., Hoang, T., Glass, L., and Sun, J. (2020b). Caster: Predicting drug interactions with chemical substructure representation. In *AAAI*.

Huang, K. and Zitnik, M. (2020). Graph meta learning via local subgraphs. *NeurIPS*.

Ioannidis, V. N., Song, X., Manchanda, S., Li, M., Pan, X., Zheng, D., Ning, X., Zeng, X., and Karypis, G. (2020). DRKG - Drug Repurposing Knowledge Graph for Covid-19. https://github.com/gnn4dr/DRKG/.

Jones, P. K., Shaw, B. H., and Raj, S. R. (2015). Orthostatic hypotension: managing a difficult problem. *Expert Review of Cardiovascular Therapy*.

Karim, M. R., Cochez, M., Jares, J. B., Uddin, M., Beyan, O., and Decker, S. (2019). Drug-drug interaction prediction based on knowledge graph embeddings and convolutional-lstm network. In *ACM-BCB*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv:1412.6980*.

Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. *ICLR*.

Liang, C., Yu, Y., Jiang, H., Er, S., Wang, R., Zhao, T., and Zhang, C. (2020). Bond: Bert-assisted open-domain named entity recognition with distant supervision. In *KDD*.

Lin, X., Quan, Z., Wang, Z.-J., Ma, T., and Zeng, X. (2020). Kgnn: Knowledge graph neural network for drug-drug interaction prediction. In *IJCAI*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *NeurIPS*.

Percha, B. and Altman, R. B. (2013). Informatics confronts drug–drug interactions. *Trends in Pharmacological Sciences*, 34(3):178–184.

Rogers, D. and Hahn, M. (2010). Extended-connectivity fingerprints. *Journal of Chemical Information and Modeling*.

Rotmensch, M., Halpern, Y., Tlimat, A., Horng, S., and Sontag, D. (2017). Learning a health knowledge graph from electronic medical records. *Scientific Reports*.

Ryu, J. Y., Kim, H. U., and Lee, S. Y. (2018). Deep learning improves prediction of drug–drug and drug–food interactions. *PNAS*.

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 *ESWC*.

Shang, J., Ma, T., Xiao, C., and Sun, J. (2019). Pre-training of graph augmented transformers for medication recommendation. In *IJCAI*.

Shaw, P., Uszkoreit, J., and Vaswani, A. (2018). Self-attention with relative position representations. In *NAACL*.

Shi, H., Yao, Q., Guo, Q., Li, Y., Zhang, L., Ye, J., Li, Y., and Liu, Y. (2020). Predicting origin-destination flow via multi-perspective graph convolutional network. In *ICDE*.

Srinivasa, R. S., Xiao, C., Glass, L., Romberg, J., and Sun, J. (2020). Fast graph attention networks using effective resistance based graph sparsification. *arXiv:2006.08796*.

Su, C., Tong, J., Zhu, Y., Cui, P., and Wang, F. (2020). Network embedding in biomedical data science. *Briefings in Bioinformatics*.

Tatonetti, N. P., Patrick, P. Y., Daneshjou, R., and Altman, R. B. (2012). Data-driven prediction of drug effects and interactions. *Science Translational Medicine*.

Teru, K. and Hamilton, W. (2020). Inductive relation prediction on knowledge graphs. In *ICML*.

Trouillon, T., Welbl, J., Riedel, S., Gaussier, E., and Bouchard, G. (2016). Complex embeddings for simple link prediction. In *ICML*.

Tzeng, R.-C. and Wu, S.-H. (2019). Distributed, egocentric representations of graphs for detecting critical structures. In *ICML*.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2018). Graph attention networks. In *ICLR*.

Veličković, P., Fedus, W., Hamilton, W. L., Liò, P., Bengio, Y., and Hjelm, R. D. (2019). Deep graph infomax. In *ICLR*.

Wang, G., Zhang, W., Wang, R., Zhou, Y., Chen, X., Zhang, W., Zhu, H., and Chen, H. (2018). Label-free distant supervision for relation extraction via knowledge graph embedding. In *EMNLP*.

Wang, H., Zhang, F., Zhang, M., Leskovec, J., Zhao, M., Li, W., and Wang, Z. (2019a). Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In *KDD*.

Wang, X., He, X., Cao, Y., Liu, M., and Chua, T.-S. (2019b). Kgat: Knowledge graph attention network for recommendation. In *KDD*.

Wishart, D. S., Knox, C., Guo, A., Cheng, D., Shrivastava, S., Tzur, D., Gautam, B., and Hassanali, M. (2008). Drugbank: a knowledgebase for drugs, drug actions and drug targets. *Nucleic Acids Research*.

Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.-i., and Jegelka, S. (2018). Representation learning on graphs with jumping knowledge networks. In *ICML*.

Yu, D., Yang, Y., Zhang, R., and Wu, Y. (2020a). Generalized multi-relational graph convolution network. *arXiv:2006.07331*.

Yu, Y., Li, Y., Shen, J., Feng, H., Sun, J., and Zhang, C. (2020b). Steam: Self-supervised taxonomy expansion with mini-paths. In *KDD*.

Yu, Y., Xia, T., Wang, H., Feng, J., and Li, Y. (2020c). Semantic-aware spatio-temporal app usage representation via graph convolutional network. *IMWUT*.

Zeng, H., Zhou, H., Srivastava, A., Kannan, R., and Prasanna, V. (2020). Graphsaint: Graph sampling based inductive learning method. In *ICLR*.

Zhang, M. and Chen, Y. (2018). Link prediction based on graph neural networks. In *NeurIPS*.

Zitnik, M., Agrawal, M., and Leskovec, J. (2018a). Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*.

Zitnik, M., Sosič, R., Maheshwari, S., and Leskovec, J. (2018b). BioSNAP Datasets: Stanford biomedical network dataset collection. http://snap.stanford.edu/biodata.

A Implementation Details

A.1 SumGNN Parameter Setup

We use the following hyperparameter set for SumGNN after random search on validation set. We use 1,024-bits Morgan fingerprint for drug featurization. We set the subgraph to be 2-hops neighbors (i.e. $k = 2$). In the subgraph summarization module, we use weight matrix of size $d = 32$ for $\mathbf{W}_1$ and $\mathbf{W}_2$. The hidden dimension $\mathbf{h}_v^k$ is set to be $d = 32$. The relation matrix $\mathbf{r}$ is set to be 32. The edge pruning threshold is set to be $\gamma = 0$. The input hidden representation of each node is $d = 32$. The number of basis B in Eq. (3) is set to 8 as the performance do not change much when set from 4 to 16 and suffer from over-fitting with $B > 16$. We study the effect of key parameter d, γ and k in our experiment part (Section 4).

A.2 Training Details

Training Parameters. For both our method and baselines, the training parameters are set as follows unless specified.

We train the model for 50 epochs with batch size 256. Our model is optimized with ADAM optimizer (Kingma and Ba, 2014) of learning rate 5×10^{-3} with gradient clipping set to 10 under L2 norm. We set the L2 weight decay to 1×10^{-5}, the layer of GNN to 2 and set the dropout rate to 0.3 for each Layer in GNN.

Model Implementation and Computing Infrastructure. All methods are implemented in PyTorch[2] and the graph neural network modules are build on Deep Graph Library (DGL)[3]. The System we use is Ubuntu 18.04.3 LTS with Python 3.6, Pytorch 1.2 and DGL 0.4.3. Our code is run in a Intel(R) Core(TM) i7-5930K CPU @ 3.50GHz CPU and a GeForce GTX TITAN X GPU.

A.3 The Range for Tuning Hyper-parameters

We use grid search to determine hyper-parameters and list the search space of key hyper-parameters as follows.

Table 3: The range for tuning hyper-parameters. The bold numbers are the default settings.

Parameters	Range
Learning Rate	$[5 \times 10^{-4}, 1 \times 10^{-3}, \mathbf{5 \times 10^{-3}}, 1 \times 10^{-2}]$
Weight Decay	$[1 \times 10^{-6}, \mathbf{1 \times 10^{-5}}, 1 \times 10^{-4}, 1 \times 10^{-3}]$
Dropout	[**0.3**, 0.4, 0.5]
Layers of GNN	[1, **2**, 3]
d	[8, 16, **32**, 64]
k	[1, **2**, 3, 4]
B	[4, **8**, 12, 16, 24, 32]

[2] https://pytorch.org/

[3] https://www.dgl.ai/

A.4 Baseline Setup

For the baselines, the settings are described as follows:

- **MLP**: We implement MLP with Pytorch with the Morgan fingerprint. We use a two-layer MLP and set the hidden dimension to 100 with dropout 0.3.
- **Node2vec**: We follow the officially released implementation from authors[4] and set the embedding dimension to 64.
- **Decagon**: We use DGL to implement the model. Following (Zitnik et al., 2018a), we set the number of GNN layers to 2 set the hidden dimension to 64 and 32 for two layers with a dropout rate of 0.1 and a minibatch size of 512.
- **GAT**: We use DGL to implement the model and set the hidden dimension to 64 with 4 attention heads, as we find that improving the number of heads will hurt the performance. We set the activation function to `LeakyReLU` with $\alpha = 0.2$.
- **Others**: We follow the officially released implementa- tion from the authors listed as follows:
  - **SkipGNN**: https://github.com/kexinhuang12345/SkipGNN.
  - **KG-DDI**: the neural model is based on code in https://github.com/rezacsedu/Drug-Drug-Interaction-Prediction, and the KG embeddings are trained via OpenKE toolbox https://github.com/thunlp/OpenKE.
  - **GraIL**: https://github.com/kkteru/grail.
  - **KGNN**: https://github.com/xzenglab/KGNN.

[4] https://github.com/aditya-grover/node2vec