scientific reports

Check for updates

OPEN

SkipGNN: predicting molecular interactions with skip-graph networks

Kexin Huang[1], Cao Xiao[2], Lucas M. Glass[2], Marinka Zitnik[3] & Jimeng Sun[4]✉

Molecular interaction networks are powerful resources for molecular discovery. They are increasingly used with machine learning methods to predict biologically meaningful interactions. While deep learning on graphs has dramatically advanced the prediction prowess, current graph neural network (GNN) methods are mainly optimized for prediction on the basis of direct similarity between interacting nodes. In biological networks, however, similarity between nodes that do not directly interact has proved incredibly useful in the last decade across a variety of interaction networks. Here, we present SkipGNN, a graph neural network approach for the prediction of molecular interactions. SkipGNN predicts molecular interactions by not only aggregating information from direct interactions but also from second-order interactions, which we call skip similarity. In contrast to existing GNNs, SkipGNN receives neural messages from two-hop neighbors as well as immediate neighbors in the interaction network and non-linearly transforms the messages to obtain useful information for prediction. To inject skip similarity into a GNN, we construct a modified version of the original network, called the skip graph. We then develop an iterative fusion scheme that optimizes a GNN using both the skip graph and the original graph. Experiments on four interaction networks, including drug–drug, drug–target, protein–protein, and gene–disease interactions, show that SkipGNN achieves superior and robust performance. Furthermore, we show that unlike popular GNNs, SkipGNN learns biologically meaningful embeddings and performs especially well on noisy, incomplete interaction networks.

Molecular interaction networks are ubiquitous in biological systems. Over the last decade, interaction networks have advanced our systems-level understanding of biology[1]. Further, they have enabled discovery of biologically significant, yet previously unmapped relationships[2], including drug–target interactions (DTIs)[3], drug–drug interactions (DDIs)[4], protein–protein interactions (PPIs)[5], and gene–disease interactions (GDIs)[6]. To assist in these discoveries, a plethora of computational methods, primarily optimized for link prediction from networks (e.g.,[7]), were developed to predict new interactions in molecular networks. Recently, deep learning on graphs has emerged as a dominant class of methods that have revolutionized state-of-the-art in learning and reasoning over network datasets. These methods, often referred to as graph neural networks (GNNs)[8] and graph convolutional networks (GCNs)[9,10], operate by performing a series of non-linear transformations on the input molecular network, where each transformation aggregates information only from immediate neighbors, i.e., direct interactors in the network. While these methods yield powerful predictors, they explicitly take into account only direct similarity between nodes in the network. Therefore, GNNs are limited at fully capturing important information for prediction that resides further away from a particular interaction in the network that we want to predict[11].

Indirect similarity between nodes that do not directly interact, e.g., the similarity in second-order interactions, has proved incredibly useful across a variety of molecular networks, including genetic interaction and protein–protein interaction networks[12–15]. This is because interactions can exist between nodes that are not necessarily similar, as illustrated in Fig. 1. For example, in a drug–target interaction (DTI) network, an edge indicates that a drug binds to a target protein. Thus, two drugs are similar because they bind to the same target protein. In contrast, a drug and a target protein are not biologically similar, although they are connected by an edge in the DTI network. This example illustrates the importance of second-order interactions, which we refer

[1]Health Data Science, Harvard T.H. Chan School of Public Health, Boston, MA, USA. [2]Analytics Center of Excellence, IQVIA, Cambridge, MA, USA. [3]Department of Biomedical Informatics, Harvard University, Boston, MA, USA. [4]Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA. ✉email: jimeng@illinois.edu

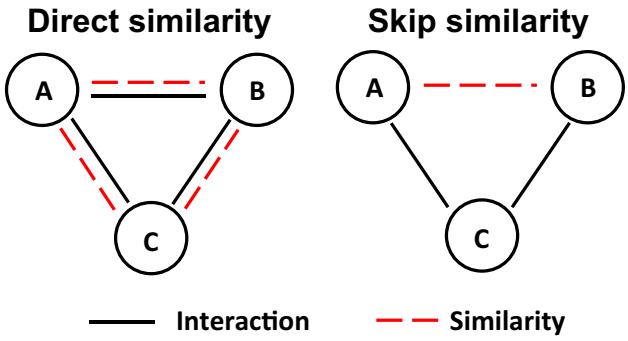

Figure 1. Direct versus skip similarity. (Left) Traditionally, an interaction between nodes A and B implies that A and B are similar and vice versa[16]. (Right) In contrast, in molecular interaction networks, directly interacting entities are not necessarily similar, which has been observed in numerous networks, including genetic interaction networks[12,13] and protein–protein interaction networks[14,15].

to as *skip similarity* (Fig. 1). For this reason, we need GNNs to predict molecular interactions, not only via direct interactions but also via similarity in second-order interactions.

Present work. Here, we present SkipGNN, a graph neural network (GNN) method for the prediction of molecular interactions. In contrast to existing GNNs, such as GCN[9], SkipGNN specifies a neural architecture, in which neural messages are passed not only via direct interactions, referred to as direct similarity, but also via similarity in second-order interactions, referred to as *skip similarity* (Fig. 1). Importantly, while the principle of *skip similarity* governs many types of molecular interaction networks, popular GNN methods fail to capture the principle. Because of that, as we show here, they cannot fully utilize molecular interaction networks. SkipGNN takes as input a molecular interaction network and uses it to construct a *skip graph*. This second-order network representation captures the *skip similarity*. SkipGNN then uses both the original graph (i.e., the input interaction network) and the skip graph to learn what is the best way to propagate and transform neural messages along edges in each graph to optimize for the discovery of new interactions.

We evaluate SkipGNN on four types of interaction networks, including two homogeneous networks, i.e., drug–drug interaction and protein–protein interaction networks, and two heterogeneous networks, i.e., drug–target interaction and gene–disease interaction networks. SkipGNN outperforms baselines that use random walks, shallow network embeddings, spectral clustering, network metrics and various state-of-the-art graph neural networks[11,15,17–21].

By examining SkipGNN 's performance in increasingly harder prediction settings when large fractions of interactions are removed from the network, we find that SkipGNN achieves robust performance. In particular, across all interaction networks, SkipGNN consistently outperforms all baseline methods, even when interaction networks are highly incomplete ("Predicting molecular interactions, Robust learning on incomplete interaction networ" section). We find that the robust performance of SkipGNN can be explained by the spectral property of skip graph, as it can preserve network structure in the face of incomplete interaction information (Supplementary D), which is also confirmed experimentally ("Ablation studies" section).

Further, we examine embeddings learned by SkipGNN and find that SkipGNN learns biologically meaningful embeddings, whereas a regular GCN does not ("SkipGNN learns meaningful embedding spaces" section). For example, when analyzing a drug–target interaction network, SkipGNN generates the embedding space in which drugs are generally separated from most of proteins while still being positioned close to the proteins to which they directly bind. Lastly, in the case of the drug–drug interaction network, we use the literature search to find evidence for SkipGNN 's novel drug–drug interaction predictions ("Investigation of SkipGNN's novel predictions" section).

Related work. Existing link prediction methods belong to one of the following categories. (1) Heuristic or mechanistic methods (e.g.,[15,22–24]) calculate an index similarity score to measure the probability of a link given the network structure around the two target nodes, such as Preferential Attachment (PA)[25] and Local Path Index (LP)[26]. However, these methods usually make strong assumptions about the network structure and hence suffer from instability of performance[15,22]. (2) Direct embedding methods generate embeddings for every node in the network capturing the node's local network topology (e.g.,[27–29]). A popular approach is to use random walks with a skip-gram model, such as DeepWalk[17], node2vec[18], and LINE[30]. The other popular approach leverages the spectral graph theory to generate a spectral embedding such as spectral clustering[20]. The generated node embeddings are then fed into a decoder classifier to predict the link existing probability. (3) Neural embedding methods, such as Graph Neural Networks (GNNs)[9,31], Variational Graph Autoencoders (VGAE)[32,33], and Graph Attention Networks (GAT)[10] use neighborhood message passing scheme to generate node embeddings and these embeddings are directly optimized in an end-to-end manner by a link prediction loss (e.g., cross-entropy). GNNs are a powerful class of models in capturing complicated graph topology. Typically, an L-layers GNN is able to propagate information of nodes in the L-hop neighborhoods[9,21]. However, the messages of nodes farther away from the central node have discounted propagation power. Thus, the vanilla GNN is limited at capturing *skip similarity*, which is from second-hop neighbors. In contrast, SkipGNN utilizes an additional skip-graph to

fully exploit this important quality for biomedical interaction network. Notably, there are recent advancements in GNN such as MixHop[11], JK-Net[34] which are designed to capture higher order graph structures through skip connections and higher order adjacency matrix. However, they are motivated by general network model and does not propose a solution for the specific challenge of 2-hop skip similarity in biomedical network.

In molecular interaction networks, the goal is to predict if a given pair of biomedical entities such as proteins, drugs or diseases will interact. We can divide methods for interaction prediction into three main groups. (1) Structural representation learning generates embeddings for each entity using the entity's structural representation, such as a compound's molecular graph or a protein's amino acid sequence. The embeddings of two entities are then combined and fed into a decoder for prediction. For example[35–37], use graph-convolutional (GCN) and convolutional (CNN) networks on molecular graphs and gene sequence data to predict binding of drugs to target proteins. Similarly[38–40], learn embedding for drugs and concatenate embeddings of drug pairs to predict drug–drug interactions. (2) Similarity-based learning is based on the assumption that entities with similar interaction patterns are likely to interact. These methods devise a similarity measure (e.g., a graphlet-based signature of proteins in the PPI network[41]) and then use the measure to predict interactions based on how similar a candidate interaction is to known interactions. A variety of techniques are used to aggregate similarity values and score interactions, including matrix factorization[42], clustering[43], and label propagation[44]. (3) Finally, network relational learning views the task as a network completion problem. It uses network structure together with side information about nodes to complete the network and predict interactions[4,33,45]. SkipGNN belongs to the structural representation learning category.

Preliminaries on graph neural networks (GNNs). Next, we describe graph neural networks as they are one of the state-of-the-art models for link prediction and are also the focus of our study. The input to a GNN is the network, represented by its adjacency matrix $\mathbf{A}$. Most often, the goal (output) of the GNN is to learn an embedding for each node in the network by capturing the network structure as well as node attributes. GNN can be represented as a series of neighborhood aggregations layers (e.g.,[9]): $\mathbf{H}^{(l+1)} = \sigma(\widetilde{\mathbf{D}}^{-\frac{1}{2}}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W})$, where $\mathbf{H}^{(l)}$ is a matrix of node embeddings at the lth layer, $\mathbf{H}^{(0)}$ are input node attributes, $\mathbf{W}$ is a trainable parameter matrix, σ is a non-linear activation function, and $\widetilde{\mathbf{D}}$ and $\widetilde{\mathbf{A}}$ are the renormalized degree and adjacency matrices, defined as: $\widetilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ and $\widetilde{\mathbf{D}}_{ii} = \sum_j \widetilde{\mathbf{A}}_{ij}$ ($\mathbf{I}$ is the identity matrix). The GNN propagates information across network neighborhoods and transforms the information in a way that is most useful for a downstream prediction tasks, such as link prediction.

Methods

SkipGNN is a graph neural network uniquely suited for molecular interactions. SkipGNN takes as input a molecular interaction network and uses it to construct a skip graph, which is a second-order network representation capturing the *skip similarity*. SkipGNN then specifies a novel graph neural network architecture that fuses the original and the skip graph to accurately and robustly predict new molecular interactions. Notations are described in Table 1.

Problem formulation. Consider an interaction network G on N nodes representing biomedical entities $\mathscr{V}$ (e.g., drugs, proteins, or diseases) and M edges $\mathscr{E}$ representing interactions between the entities. For example, G can be a drug–target interaction network recording information on how drugs bind to their protein targets[3]. For every pair of entities i and j, we denote their interaction with a binary indicator $e_{ij} \in \{0, 1\}$, indicating the experimental evidence that i and j interact (i.e., $e_{ij} = 1$) or the absence of evidence for interaction (i.e., $e_{ij} = 0$). We denote the adjacency matrix of G as $\mathbf{A}$, where $\mathbf{A}_{ij}$ is 1 if nodes i and j are connected ($e_{ij} = 1$) in the graph and otherwise 0 ($e_{ij} = 0$). Further, $\mathbf{D}$ is the degree matrix, a diagonal matrix, where $\mathbf{D}_{ii}$ is the degree of node i.

Problem (Molecular Interaction Prediction) *Given a molecular interaction network $G = (\mathscr{V}, \mathscr{E})$, we aim to learn a mapping function $f : \mathscr{E} \rightarrow [0, 1]$ from edges to probabilities such that $f(i, j)$ optimizes the probability that nodes i and j interact.*

Construction of the skip graph. Next, we describe skip graphs, the key novel representation of interaction networks that allow for effective use of GNNs for predicting interactions. We realize *Skip similarity* by encouraging the GNN model to embed skipped nodes close together in the embedding space. To do that, we construct skip graph G_s, in two-hop neighbors are connected by edges. This construction creates paths in G_s along which neural messages can be exchanged between the skipped nodes.

Formally, we use the following operator to obtain the skip graph's adjacency matrix $\mathbf{A}_s$:

$$\mathbf{A}_s^{ij} = \begin{cases} 1 \text{ if } \exists \text{ k s.t. (i, k)} \in \mathscr{E} \text{ and (k, j)} \in \mathscr{E} \\ 0 \text{ otherwise.} \end{cases}$$

The corresponding degree matrix is $\mathbf{D}_s^{ii} = \sum_j \mathbf{A}_s^{ij}$. An efficient way to implement the skip graph is through matrix multiplication:

$$\mathbf{A}_s = \text{sign}(\mathbf{A}\mathbf{A}^{\mathrm{T}}), \tag{1}$$

where sign(x) is the sign function, sign(x) = 1 if $x > 0$ and 0 otherwise, which is applied element-wise on $\mathbf{A}\mathbf{A}^{\mathrm{T}}$. It counts the number of two-hop paths from node i to j. Hence, if an entry for node i, j in $\mathbf{A}\mathbf{A}^{\mathrm{T}}$ is larger than 0, it means there exists a skipped node between node i, j. Then, we convert the positive entry into 1 to construct the skip graph's adjacent matrix. Given this skip graph, we proceed to describe the full SkipGNN model.

Notation	Definition
$G:\{\mathscr{V},\mathscr{E}\}$	Graph with nodes $\mathscr{V}$ and edges $\mathscr{E}$
$\mathbf{D},\mathbf{A}\in\mathbb{N}^{N\times N}$	Degree and adjacency matrices for graph G
$\widetilde{\mathbf{D}},\widetilde{\mathbf{A}}\in\mathbb{N}^{N\times N}$	Normalized degree and adjacency matrices for G
$\mathbf{X}\in\mathbb{R}^{N\times D}$	D-dimensional node embeddings
$e_{ij}\in\{0,1\}$	Ground-truth interaction between nodes i and j
G_s	Skip graph
$\mathbf{D}_s,\mathbf{A}_s\in\mathbb{N}^{N\times N}$	Degree and adjacency matrices for G_s
$\widetilde{\mathbf{D}_s},\widetilde{\mathbf{A}_s}\in\mathbb{R}^{N\times N}$	Normalized degree and adjacency matrices for G_s
$\mathbf{H}^{(l)},\mathbf{S}^{(l)}$	Node embeddings for G and G_s, in layer l
$\mathbf{E}$	Final node embeddings
$p_{ij}\in[0,1]$	Probability of interaction between nodes i and j
$y_{ij}\in\{0,1\}$	Binary indicator of interaction between nodes i and j
$\mathscr{L}\in\mathbb{R}$	Binary cross-entropy loss
$\mathbf{W}_o^{(l)},\mathbf{W}_s^{(l)}$	Weight matrix for original (o) and skip (s) graphs, layer l
$\mathbf{W}_o^{'(l)}$	Weight matrix for skip-to-original-graph fusion
$\mathbf{W}_s^{'(l)}$	Weight matrix for original-to-skip-graph fusion
$\mathbf{W}_d,b$	Decoder weight matrix and bias parameter

Table 1. Notation used in SkipGNN.

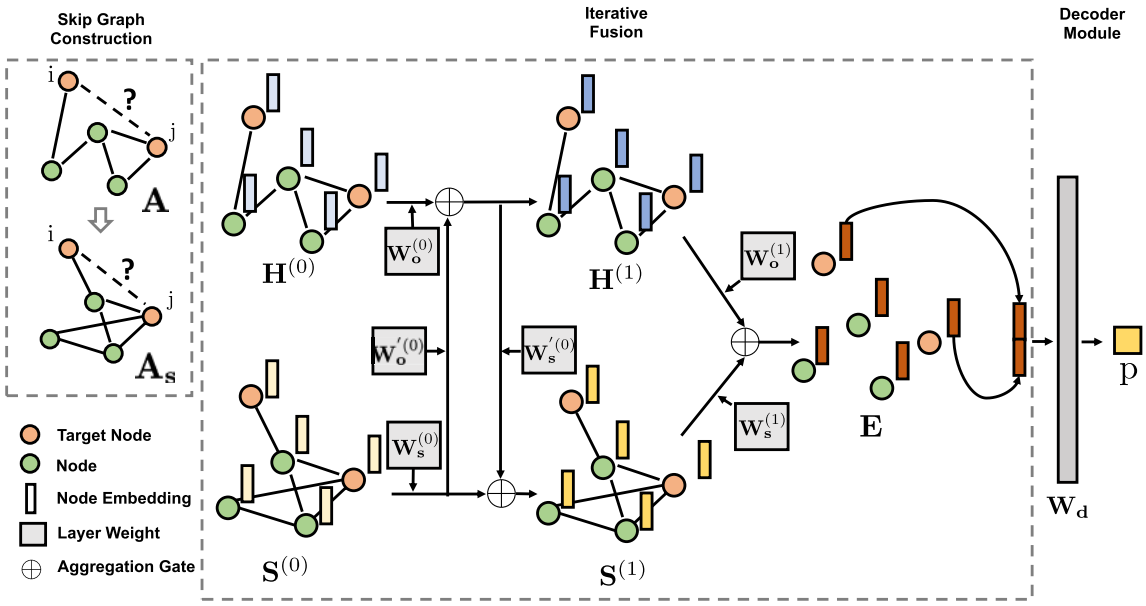

Figure 2. Neural architecture of SkipGNN. (Left) SkipGNN constructs skip graph G_s (denoted by adjacency matrix $\mathbf{A}_s$) based on the input graph G (denoted by adjacency matrix $\mathbf{A}$) using Eq. (1). (Middle) Initial node embeddings, $\mathbf{H}^{(0)}$ and $\mathbf{S}^{(0)}$, are specified using side information (e.g., gene expression vectors if nodes represent genes) or generated using node2vec[18]. In SkipGNN, node embeddings are then propagated along edges of G_s and G and transformed through a series of computations (layers), which output powerful embeddings that can then be used for downstream prediction of interactions. Illustrated is a two-layer iterative fusion scheme. In the first layer, two GNNs with parameter weight matrices $\mathbf{W}_o^{(0)}$ and $\mathbf{W}_s^{(0)}$ (operating on $\mathbf{A}$ and $\mathbf{A}_s$, respectively) are fused via weight matrices $\mathbf{W}_o^{'(0)}$ and $\mathbf{W}_s^{'(0)}$ based on Eq. (2). This completes computations in the first layer of SkipGNN, producing embeddings $\mathbf{H}^{(1)}$ and $\mathbf{S}^{(1)}$. In the second layer, those embeddings are transformed via $\mathbf{W}_o^{(1)}$ and $\mathbf{W}_s^{(1)}$ using Eq. (3), resulting in final embeddings $\mathbf{E}$. (Right) Embeddings $\mathbf{E}_i$ and $\mathbf{E}_j$ of target nodes i and j are retrieved, concatenated, and then fed into a decoder (parameterized by $\mathbf{W}_d$). Decoder returns p_{ij}, representing the probability that nodes i and j interact.

The SkipGNN model. In this section, we describe how we leverage the skip graph for link prediction. After we generate the novel skip graph from "Construction of the skip graph" section, we propose an iterative fusion scheme for SkipGNN to allow the skip graph and the original graph to learn from each other for better integration. Lastly, a decoder is used to output a probability measuring if the given pair of molecular entities interact.

Iterative fusion. We want a model to automatically learn how to balance between *direct similarity* and *skip similarity* in the final embedding. We design an iterative fusion scheme with aggregation gates to combine both similarity information. The motivation is that to represent biomedical entity to its fullest extent, node embedding must capture its complicated bioactive functions with *skip/direct similarities*. Hence, instead of simply concatenating the output node embeddings from the GNN output of the original graph G that captures *direct similarity* and skip graph G_s that captures *skip similarity*, we allow two GNNs on G and G_s to interact with each other iteratively via the following propagation rules (see Fig. 2):

$$\mathbf{H}^{(l+1)} = \sigma(\mathrm{AGG}(\mathbf{F}\mathbf{H}^{(l)}\mathbf{W}_o^{(l)}, \mathbf{F}_s\mathbf{S}^{(l)}\mathbf{W}_o^{'(l)})), \quad \mathbf{S}^{(l+1)} = \sigma(\mathrm{AGG}(\mathbf{F}_s\mathbf{S}^{(l)}\mathbf{W}_s^{(l)}, \mathbf{F}\mathbf{H}^{(l+1)}\mathbf{W}_s^{'(l)})), \tag{2}$$

where $\mathbf{F} = \widetilde{\mathbf{D}}^{-\frac{1}{2}}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-\frac{1}{2}}$, $\mathbf{F}_s = \widetilde{\mathbf{D}}_s^{-\frac{1}{2}}\widetilde{\mathbf{A}}_s\widetilde{\mathbf{D}}_s^{-\frac{1}{2}}$. Here, $\mathbf{H}^{(l)}, \mathbf{S}^{(l)}$ are node embeddings at the lth layer from direct similarity graph G and *skip similarity* graph G_S, respectively. $\mathbf{F}, \mathbf{F}_s$ are the re-normalized adjacency matrices from direct similarity and *skip similarity*, respectively. And $\mathbf{W}_o^{(l)}, \mathbf{W}_o^{'(l)}, \mathbf{W}_s^{(l)}, \mathbf{W}_s^{'(l)}$ are the transformed weights for layer l. $\mathbf{H}^{(0)}$ and $\mathbf{S}^{(0)}$ are set to be $\mathbf{X}$, the input node attributes generated from node2vec. The aggregate gate AGG in Eq. (2) can be a summation, a Hadamard product, max-pooling, or some other aggregation operator[46]. Empirically, we find that summation gate has the best performance. $\sigma()$ is the activation function and we use $\mathrm{ReLU}(\cdot) = \max(\cdot, 0)$ to add non-linearity in the propagation.

In each iteration, the node embedding for original graph $\mathbf{H}^{(l+1)}$ is first updated with its previous layer's node embedding $\mathbf{H}^{(l)}$, combined with skip graph embedding $\mathbf{S}^{(l)}$. After obtaining the updated original graph embedding $\mathbf{H}^{(l+1)}$, we then update the skip graph embedding $\mathbf{S}^{(l+1)}$ in a similar fashion.

This update rule is very different from simple concatenation as it is an iterative process where each update of the node embedding for each graph is affected by the most *recent* node embedding from both graphs. This way, two embedding are learned to find the best dependency structure between each other and fuse into one final embedding instead of a simple concatenation. In the last layer, final node embedding $\mathbf{E}$ is obtained through:

$$\mathbf{E} = \mathrm{AGG}(\mathbf{F}\mathbf{H}^{(1)}\mathbf{W}_o^{(1)}, \mathbf{F}_s\mathbf{S}^{(1)}\mathbf{W}_s^{(1)}), \tag{3}$$

where (1) is the index for the last layer and AGG is the summation gate. As in the motivation, we are interested only in up to second order neighbors, thus we use two layers GNN, see Fig. 2. We don't use activation function here as it does not require an extra non-linear transformation to be fed into the decoder network. Empirically, we show this fusion scheme boosts predictive performance in "Ablation studies" section.

SkipGNN decoder. Given the target nodes (i, j) and their corresponding node embedding $\mathbf{E}_i, \mathbf{E}_j$, we implement a neural network as a decoder to first combine $\mathbf{E}_i, \mathbf{E}_j$ to obtain an input embedding through a COMB function (e.g., concatenation, sum, Hadamard product). Then, the combined embedding is fed into a neural network parametrized by weight $\mathbf{W}_d$ and bias b as a binary classifier to obtain probability p_{ij}:

$$p_{ij} = \sigma(\mathbf{W}_d\mathrm{COMB}(\mathbf{E}_i, \mathbf{E}_j) + b), \tag{4}$$

where p_{ij} represents the probability that nodes i and j interact (i.e., $f(i, j)$). We use concatenation as the COMB function as it consistently yield the best performance across different types of networks.

Algorithm 1: The SkipGNN Algorithm

Input: interaction network G with adjacent matrix $\mathbf{A}$
Node Embedding Generation, *e.g.*:
$\mathbf{X} \leftarrow \mathrm{node2vec}(\mathbf{A})$
Skip Graph Construction (Section 2.1):
$\mathbf{A}_s \leftarrow \mathrm{SkipGraph}(\mathbf{A})$ via Eq. (1)
for $t = 1 \ldots T_{\max}$ **do**
  sample mini-batch of training node pairs $\mathscr{M} \subseteq \mathscr{E}$ with corresponding labels y
  Iterative Fusion (Section 2.2.1):
  $\mathbf{H}^{(1)} \leftarrow \mathrm{FusionGate}(\mathbf{H}^{(0)})$ via Eq. (2)
  $\mathbf{S}^{(1)} \leftarrow \mathrm{FusionGate}(\mathbf{S}^{(0)})$ via Eq. (2)
  $\mathbf{E} \leftarrow \mathrm{FuseGate}(\mathbf{H}^{(1)}, \mathbf{S}^{(1)})$ via Eq. (3)
  Decoder (Section 2.2.2):
  $p_{ij} \leftarrow$ **decode**$(\mathbf{E})$ via Eq. (4)
  Compute the loss value $\mathscr{L}$ using p_{ij} and y (Section 2.3) and update model parameters via gradient descent

Dataset	Prediction task	# nodes	# edges	A
DTI	Drug–target interaction	7343	15,139	4.12
DDI	Drug–drug interaction	1514	48,514	64.09
PPI	Protein–protein interaction	5604	23,322	8.32
GDI	Gene–disease interaction	19,783	81,746	8.26

Table 2. Data statistics. 'A' indicates average node degree.

The SkipGNN algorithm. The overall algorithm is shown in Algorithm 1. Here, we only leverage accessible network information (adjacent matrix $\mathbf{A}$ of the network G) to predict links. In all experiments, we initialize embeddings using node2vec[18] as: $\mathbf{X} = \text{node2vec}(\mathbf{A})$.

Second, we construct the skip graph with adjacent matrix $\mathbf{A}_s$ via Eq. (1) to capture the *skip-similarity* principle. Next, at every step, a mini-batch of interaction pairs $\mathscr{M}$ with labels y is sampled. Then, two graph convolutions networks are used for the original graph and the skip graph respectively. In the propagation step, we use iterative fusion (Eq. (2)) to naturally combine embeddings convolved on the original graph and on the skip graph, corresponding to *direct* and *skip similarity*, respectively. In the last layer, embeddings are stored in $\mathbf{E}$. We then retrieve the embeddings for each node in the mini-batched pairs $\mathscr{M}$ and concatenate them to feed into decoder (Eq. (4)).

During training, we optimize the SkipGNN 's parameters $\mathbf{W}_o^{(l)}$, $\mathbf{W}_o^{'(l)}$, $\mathbf{W}_s^{(l)}$, $\mathbf{W}_s^{'(l)}$, $\mathbf{W}_d$, b in an end-to-end manner through a binary cross-entropy loss: $\mathscr{L} = \sum_{(i,j)\in\mathscr{M}} y_{ij} \log \mathrm{p_{ij}} + (1 - \mathrm{y_{ij}}) \log (1 - \mathrm{p_{ij}})$, where y_{ij} is the true label for nodes i and j that are sampled during training via mini-batching, $(i, j) \in \mathscr{M}$, and $\mathscr{M}$ is a mini-batch of interaction pairs. After the model is trained, it can be used to make predictions. Given two entities i and j, the model predicts probability $f(i, j)$ that i and j interact.

Results

We conduct a variety of experiments to investigate the predictive power of SkipGNN ("Predicting molecular interactions" section). We then study the method's robustness to noise and missing data ("Robust learning on incomplete interaction networ" section) and demonstrate the skip similarity principle ("SkipGNN learns meaningful embedding spaces" section). Next, we conduct ablation studies to examine contributions of each of SkipGNN 's components towards the final SkipGNN performance ("Ablation studies" section). Finally, we investigate novel predictions made by SkipGNN ("Investigation of SkipGNN's novel predictions" section).

Data and experimental setup. Next we provide details on molecular interaction datasets, baseline methods, and experimental setup.

Molecular interaction networks. We consider four publicly-available network datasets. (1) *BIOSNAP-DTI*[47] contains 5,018 drugs that target 2,325 protein through 15,139 drug–target (DTI) interactions. (2) *BIOSNAP-DDI*[47] consists of 48,514 drug–drug interactions (DDIs) between 1,514 drugs extracted from drug labels and biomedical literature. (3) *HuRI-PPI*[48] is the human reference protein–protein interaction network generated by multiple orthogonal the high-throughput yeast two-hybrid screens. We use HI-III network, which has 5,604 proteins and 23,322 interactions. (4) Finally, we consider *DisGeNET-GDI*[49] collects curated gene–disease interactions (GDIs) from GWAS studies, animal models and scientific literature. The dataset has 81,746 interactions between 9,413 genes and 10,370 diseases. Dataset statistics are described in Table 2.

SkipGNN implementation and hyperparameters. We implemented SkipGNN using PyTorch deep learning framework (The source code implementation of SkipGNN is available at https://github.com/kexinhuang12345/SkipGNN). We use a server with 2 Intel Xeon E5-2670v2 2.5GHZ CPUs, 128GB RAM and 1 NVIDIA Tesla P40 GPU. We set optimization parameters as follows: learning rate is 5e−4 using the Adam optimizer[50], mini-batch size is $|\mathscr{M}| = 256$, epoch size is 15, and dropout rate is 0.1. We set hyper-parameters using 10 runs random search based on best average prediction performance on validation set of DTI task. We find the setup is robust in other datasets. The ranges of hyper-parameters are set as follows: learning rate: [1e−3, 5e−4, 1e−4, 5e−5]; mini-batch size [32, 64, 128, 256, 512]; dropout rate [0, 0.05, 0.1, 0.2]; hidden size [16, 32, 64, 128]. Specifically, we set hidden size in the first layer as $d^{(1)} = 64$ and hidden size in the second layer as $d^{(2)} = 16$.

Baseline methods. We compare SkipGNN to seven powerful predictors of molecular interactions from network science and graph machine-learning fields. From machine learning, we use three direct network embedding methods: **DeepWalk**[17], **node2vec**[18], and we also include **struc2vec**[19]. The latter method is conceptually distinct by leveraging local network structural information, while the former methods use random walks to learn embeddings for nodes in the network. Further, we examine five graph neural networks: **VGAE**[32], **GCN**[9], **GIN**[21], **JK-Net**[34] and **MixHop**[11]. They all use the same input encoding as SkipGNN. From network science, we consider **Spectral Clustering**[20]. We also use **L3**[15] heuristic, which was recently shown to outperform over 20 network science methods for the problem of PPI prediction. Further details on baseline methods, their implementation and parameter selection are in supplementary.

Experimental setup. In all our experiments, we follow an established evaluation strategy for link prediction (e.g.,[4,51]). We divide each dataset into train, validation, and test sets in a 7:1:2 ratio, which yields positive examples (molecular interactions). We generate negative counterparts by sampling the complement set of positive examples. The cardinality of negative samples are set to be the same as positive data points. For every experiment, we conduct five independent runs with different random splits of the dataset. We select the best performing model based on the loss value on the validation set. The performance of selected model is calculated on the test set. To calculate prediction performance, we use: (1) area under precision-recall curve (PR-AUC): PR-AUC $= \sum_{k=1}^{n} \text{Prec(k)}\Delta\text{Rec(k)}$, where k is kth precision/recall operating point (Prec(k), Rec(k)); and (2) area under the receiver operating characteristics curve (ROC-AUC): ROC-AUC $= \sum_{k=1}^{n} \text{TP(k)}\Delta\text{FP(k)}$, where k is kth true-positive and false-positive operating point (TP(k), FP(k)). Higher values of PR-AUC and ROC-AUC indicate better predictive performance. In addition to the PR-AUC and ROC-AUC, we rank each method in each dataset based on its PR-AUC and provide the average rank of a method across four datasets. The rank suggests the overall performance of the method compared to others. To further show the performance gain of SkipGNN, we resort to statistical test. For each method, we take the ROC-AUC and PR-AUC of each run for each dataset as the data samples. Then, we compute the p value for Wilcoxon signed-rank test between SkipGNN and the compared method.

Predicting molecular interactions. We start by evaluating SkipGNN on four distinct types of molecular interactions, including drug–target interactions, drug–drug interactions, protein–protein interactions, and gene–disease interactions, and we then compare SkipGNN 's performance to baseline methods.

In each interaction network, we randomly mask 30% interactions as the holdout validation (20%) and test (10%) sets. The remaining 70% interactions are used to train the SkipGNN and each of the baselines. After training, each method is asked to predict whether pairs of entities in the test set will likely interact.

We report results in Table 3 and the method rank, along with the p values for statistical test are provided in Table 4. We see that SkipGNN is the top performing method out of 11 methods across all molecular interaction networks. SkipGNN has the best predictive performance for DTI and PPI datasets and has the second best performance in DDI and GDI datasets, with an average rank of 1.5. In contrast, the best performing baseline MixHop has average rank of 2.5, as it sometimes is worse than JK-Net and GIN. We also see that SkipGNN's improvement over all baselines is statistically significant ($< .05$). To show the usefulness of skip graph, we compare with GCN-backend baselines GCN and VGAE. We see up to 2.7% improvement of SkipGNN over GCN and up to 8.8% improvement over VGAE on PR-AUC. Since GCN and VGAE can only use *direct similarity*, this finding provides evidence that considering *skip similarity* and *direct similarity* together, as is made possible by SkipGNN, is important to be able to accurately predict a variety of molecular interactions. Compared to direct embedding methods, SkipGNN has up to 28.8% increase over DeepWalk, 20.4% increase over node2vec, and 15.6% over spectral clustering on PR-AUC. These results support previous observations[4] that graph neural networks can learn more powerful network representations than direct embedding methods. Finally, all baselines vary in performance across datasets/tasks while SkipGNN consistently yields the most powerful predictor.

Robust learning on incomplete interaction networks. Next, we test SkipGNN 's performance on incomplete interaction networks. Due to knowledge gaps in biology, many of today's interaction networks are incomplete and thus it is crucial that methods are robust and able to perform well even when many interactions are missing.

In this experiment, we let each method be trained on 10%, 30%, 50%, and 70% of edges in the DTI, DDI, and PPI datasets and predict on the rest of the data (we use 10% of test edges as validation set for early stopping).

Results in Fig. 3 show that SkipGNN gives the most robust results among all the methods. In all tasks, SkipGNN achieves strong performance even when having access to only 10% of the interactions. Further, in almost every percentage point, SkipGNN is better than the baselines. In addition, we see that VGAE is not robust as its performance dropped to around 0.5 PR-AUC in highly-incomplete settings on DTI and DDI tasks. Performance of node2vec and GCN steadily improve as the percentage of seen edges increases. Further, while spectral clustering is robust to incomplete data, its performance varies substantially with tasks. We conclude that SkipGNN is robust and is especially appropriate for data-scarce networks.

SkipGNN learns meaningful embedding spaces. Next, we visualize embeddings learned by GCN and SkipGNN in an effort to investigate whether SkipGNN can better capture the structure of interaction networks than GCN. For that, we use DTI and GDI networks in which drugs/diseases are linked to associated proteins/genes. We use t-SNE[52] and visualize the learned embeddings in Fig. 4 (DTI network) and Fig. 5 (GDI network). Note that both GCN and SkipGNN uses the same input embedding, which means the only difference is whether or not skip similarity is used.

First, we observe that GCN cannot distinguish between different types of biomedical entities (i.e., drugs vs. proteins and disease vs. genes). In contrast, SkipGNN can successfully separate the entities, as evidenced by distinguishable groups of points of the same color in the t-SNE visualizations. This observation confirms that SkipGNN has a unique ability to capture the *skip similarity* whereas GCN cannot. This is because GCN forces embeddings of connected drug-protein/gene–disease pairs to be similar and thus it embeds those pairs close together in the embedding space. However, by doing so, GCN conflates drugs with proteins and genes with diseases. In contrast, SkipGNN generates a biologically meaningful embedding space in which drugs are distinguished from proteins (or, genes from diseases) while drugs are still positioned in the embedding space close to proteins to which they bind (or, in the case of GDI network, diseases are positioned close to relevant disease-associated genes).

Task	Method	PR-AUC	ROC-AUC	Rank
DTI	DeepWalk	0.753 ± 0.008	0.735 ± 0.009	10
	node2vec	0.771 ± 0.005	0.720 ± 0.010	9
	struc2vec	0.677 ± 0.007	0.656 ± 0.010	11
	SC	0.818 ± 0.007	0.743 ± 0.008	8
	L3	0.891 ± 0.004	0.793 ± 0.006	6
	VGAE	0.853 ± 0.010	0.800 ± 0.010	7
	GCN	0.904 ± 0.011	0.899 ± 0.010	5
	GIN	0.922 ± 0.004	0.907 ± 0.006	3
	JK-Net	0.921 ± 0.006	0.907 ± 0.008	4
	MixHop	0.921 ± 0.006	0.920 ± 0.004	2
	SkipGNN	**0.928 ± 0.006**	**0.922 ± 0.004**	1
DDI	DeepWalk	0.698 ± 0.012	0.712 ± 0.009	10
	node2vec	0.801 ± 0.004	0.809 ± 0.002	8
	struc2vec	0.643 ± 0.012	0.654 ± 0.007	11
	SC	0.749 ± 0.009	0.816 ± 0.006	9
	L3	0.860 ± 0.004	0.869 ± 0.003	4
	VGAE	0.844 ± 0.076	0.878 ± 0.008	7
	GCN	0.856 ± 0.005	0.875 ± 0.004	5
	GIN	0.856 ± 0.005	0.876 ± 0.003	5
	JK-Net	**0.870 ± 0.009**	0.885 ± 0.005	1
	MixHop	0.861 ± 0.006	0.879 ± 0.004	3
	SkipGNN	0.866 ± 0.006	**0.886 ± 0.003**	2
PPI	DeepWalk	0.715 ± 0.008	0.706 ± 0.005	11
	node2vec	0.773 ± 0.010	0.766 ± 0.005	10
	struc2vec	0.875 ± 0.004	0.868 ± 0.006	8
	SC	0.897 ± 0.003	0.859 ± 0.003	7
	L3	0.899 ± 0.003	0.861 ± 0.003	6
	VGAE	0.875 ± 0.004	0.844 ± 0.006	8
	GCN	0.909 ± 0.002	0.907 ± 0.006	4
	GIN	0.907 ± 0.004	0.897 ± 0.006	5
	JK-Net	0.912 ± 0.003	0.901 ± 0.005	3
	MixHop	0.909 ± 0.004	0.913 ± 0.003	2
	SkipGNN	**0.921 ± 0.003**	**0.917 ± 0.004**	1
GDI	DeepWalk	0.827 ± 0.007	0.832 ± 0.003	11
	node2vec	0.828 ± 0.006	0.834 ± 0.003	10
	struc2vec	0.910 ± 0.006	0.909 ± 0.005	4
	SC	0.905 ± 0.002	0.863 ± 0.003	6
	L3	0.899 ± 0.001	0.832 ± 0.001	8
	VGAE	0.902 ± 0.006	0.873 ± 0.009	7
	GCN	0.909 ± 0.002	0.906 ± 0.006	5
	GIN	**0.916 ± 0.004**	0.900 ± 0.005	1
	JK-Net	0.891 ± 0.049	0.898 ± 0.002	9
	MixHop	0.912 ± 0.005	**0.916 ± 0.004**	3
	SkipGNN	0.915 ± 0.003	0.912 ± 0.004	2

Table 3. Predictive performance. SkipGNN achieves the best performance across all metrics and tasks compared to baselines. Results of five independent runs on DDI, PPI, DTI and GDI tasks on state of the art link prediction algorithms.

Method	DeepWalk	node2vec	struc2vec	SC	L3	VGAE	GCN	GIN	JK-Net	MixHop	SkipGNN
Average Rank	10.5	9.25	8.50	7.50	6.00	7.25	4.75	3.75	4.00	2.50	1.50
p value	< .001	< .001	.006	.003	.016	.005	.012	.017	.025	.042	N/A

Table 4. Predictive performance ranking and statistical testing. We rank each tested method's PR-AUC in each dataset and computes the average rank and also computes the performance difference from SkipGNN using Wilcoxon signed-rank test. SkipGNN has the highest rank compared with 10 other baselines and its performance gain is statistically significant.

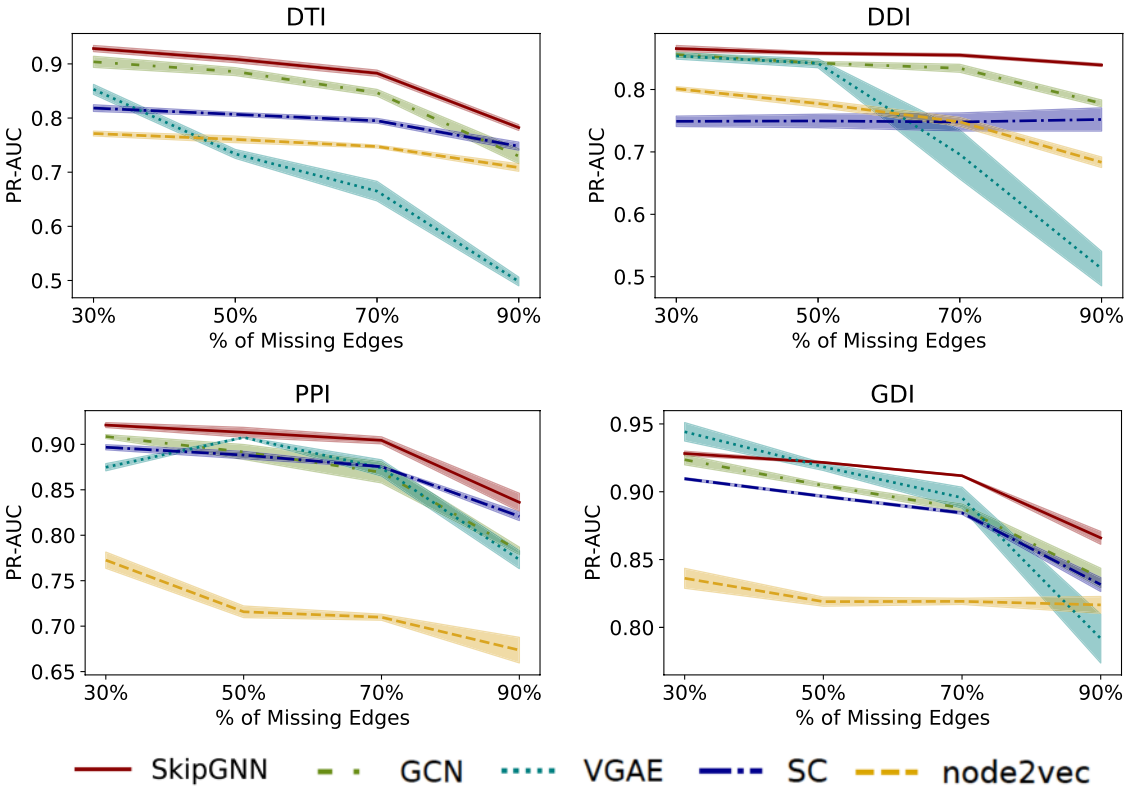

Figure 3. Predictive performance as a function of network incompleteness. SkipGNN provides robust result in varying fraction of missing edges. Fivefold average with 95% confidence interval for PR-AUC against various fractions of missing edges on four prediction tasks: drug–target interaction prediction (DTI), drug–drug interaction prediction (DDI), protein–protein interaction prediction (PPI) and gene–disease interaction prediction (GDI) on node2vec, Spectral Clustering (SC), Variational Graph Auto-Encoder (VGAE), Graph Convolutional Network (GCN), and SkipGNN. We omit DeepWalk as it has similar performance as node2vec. SkipGNN consistently shows the best performance even when networks are highly incomplete.

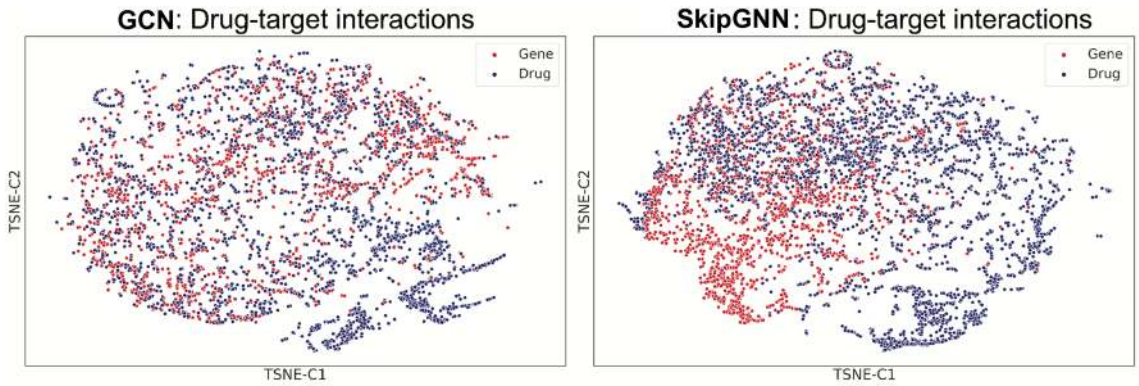

Figure 4. Visualizations of drug–target interaction network. GCN does not distinguish drug and target gene as it only captures direct similarity whereas SkipGNN is able to distinct drug and target gene embeddings, confirming its ability to capture *skip similarity*. We use GCN and SkipGNN on the drug–target interaction dataset to learn drug/target embeddings, which are visualized using t-SNE.

We also calculate the silhouette score of the t-SNE plot, which measures the inter-cluster and intra-cluster distance and is used to calculate the goodness of a clustering technique. A higher value indicates that the sample is better matched to its own cluster and poorly matched to neighboring clusters. Here SkipGNN has a silhouette score of 0.114 for DTI whereas GCN has a score of 0.014 for DTI. For GDI, SkipGNN has a score 0.079 and GCN has a score 0.018. The up to 8 times increase in silhouette scores suggest that SkipGNN can better distinguish the entities than GCN.

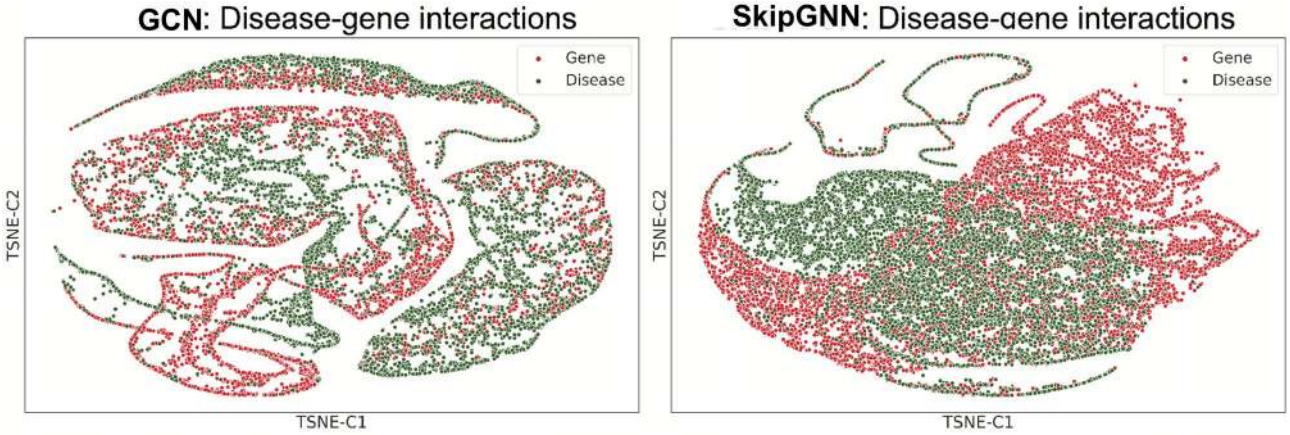

Figure 5. Visualizations of gene–disease interaction network. GCN does not distinguish disease and gene as it only captures direct similarity whereas SkipGNN is able to distinct disease and gene embeddings, confirming its ability to capture skip similarity. We use GCN and SkipGNN on the gene–disease interaction dataset to learn gene/disease embeddings, which are visualized using t-SNE.

Further, we find that GCN and its graph convolutional variants cannot capture *skip similarity* because they aggregate neural messages only from direct (i.e., immediate) neighbors in the interaction network. SkipGNN solves this problem by passing and aggregating neural message from direct as well as in-direct neighbors, thereby explicitly capturing *skip similarity*.

Ablation studies. To show that each component of SkipGNN has an important role in the final performance of SkipGNN, we conduct a series of ablation studies. SkipGNN has four key components, and we study how the metho performance changes when we remove each of the components:

- **-fusion** replaces SkipGNN 's fusion scheme with a simple concatenation of node embeddings generated by GCN.
- **-skipGraph** removes skip graph and degenerates to GCN.
- **-Weighted-L1** uses weighted-L1 gate in Eq. (2) as $\mathrm{AGG}(\mathrm{A},\mathrm{B}) = |\mathrm{A} - \mathrm{B}|$, where $|\cdot|$ is the absolute value operator.
- **-Hadamard** replaces the summation gate with Hadamard operator '$*$' in Eq. (2) such that $\mathrm{AGG}(\mathrm{A},\mathrm{B}) = \mathrm{A} * \mathrm{B}$.

Table 5 show results of deactivating each of these components, one at a time. We find that -fusion outperforms -skipGraph (i.e., GCN) by a large margin. This finding identifies skip graph as a key driver of performance improvement. Further, we find that our iterative fusion scheme is important, indicating that successful methods need to integrate both direct and *skip similarity* in interaction networks. Next, we see that weighted L_1 gate has comparable or worse performance than the summation gate and Hadamard operator performs the worst, suggesting that SkipGNN 's summation gate is the best-performing aggregation function. Altogether, we conclude that all SkipGNN 's components are necessary for its strong performance.

Investigation of SkipGNN 's novel predictions. The main goal of link prediction on graphs is to find novel hits that do not exist in the dataset. We conduct a literature search and find SkipGNN is able to discover novel hits. We select pairs that are not interacted in the original dataset but are flagged as interaction from our model. We then pick the top 10 confident interactions and feed them into literature database and see if there are evidence supporting our findings. We find promising result for the DDI task (Table 6). Out of the 10 top-ranked interaction pairs, we are able to find 6 pairs that have literature evidence support.

For example, for the interaction between Warfarin and Calozapine[53], reports that "*Clozapine* increase the concentrations of commonly used drugs in elderly like digoxin, heparin, phenytoin and *Warfarin* by displacing them from plasma protein. This can lead to increase in respective adverse effects with these medications." Also, the manufacturer[59] also reports that "*Clozapine* may displace *Warfarin* from plasma protein-binding sites. Increased levels of unbound *Warfarin* could result and could increase the risk of hemorrhage." Take another example between Warfarin and Ivacaftor[54], conducts a DDI study and reports that "caution and appropriate monitoring are recommended when concomitant substrates of CYP2C9, CYP3A and/or P-gp are used during treatment with *Ivacaftor*, particularly drugs with a narrow therapeutic index, such as *Warfarin*." Finally, we provide the top 10 outputs for DTI, PPI, and GDI tasks in Appendix 3.

Task	Method	PR-AUC	ROC-AUC
DTI	SkipGNN	**0.928 ± 0.006**	0.922 ± 0.004
	-fusion	0.909 ± 0.011	0.907 ± 0.013
	-skipGraph	0.904 ± 0.011	0.899 ± 0.010
	-Weighted-L1	0.927 ± 0.013	**0.926 ± 0.011**
	-Hadamard	0.796 ± 0.116	0.795 ± 0.116
DDI	SkipGNN	**0.866 ± 0.006**	**0.886 ± 0.003**
	-fusion	0.864 ± 0.007	0.884 ± 0.002
	-skipGraph	0.856 ± 0.005	0.875 ± 0.004
	-Weighted-L1	0.863 ± 0.006	0.885 ± 0.003
	-Hadamard	0.833 ± 0.054	0.883 ± 0.003
PPI	SkipGNN	**0.921 ± 0.003**	**0.917 ± 0.004**
	-fusion	0.912 ± 0.004	0.906 ± 0.005
	-skipGraph	0.909 ± 0.002	0.907 ± 0.006
	-Weighted-L1	0.917 ± 0.003	0.908 ± 0.006
	-Hadamard	0.909 ± 0.025	0.914 ± 0.010
GDI	SkipGNN	**0.915 ± 0.003**	**0.912 ± 0.004**
	-fusion	0.896 ± 0.029	0.892 ± 0.014
	-skipGraph	0.909 ± 0.002	0.906 ± 0.006
	-Weighted-L1	0.913 ± 0.009	0.898 ± 0.010
	-Hadamard	0.883 ± 0.041	0.891 ± 0.025

Table 5. Results of ablation experiments. SkipGNN's model components setup achieve the best result. Ablation study result of five independent runs on DDI, PPI and DTI tasks.

Rank	Drug 1	Drug 2	Evidence for DDI
1	Warfarin	Clozapine	Mukku et al., 2018[53]
2	Warfarin	Ivacaftor	Robertson et al., 2015[54]
3	Phenelzine	Deferasirox	
4	Warfarin	Paraldehyde	DuPont, Product Information[55]
5	Warfarin	Cyclosporine	Snyder, 1988[56]
6	Phenytoin	Sipuleucel-T	
7	Warfarin	Netupitant	
8	Phenelzine	Suvorexant	Merck, Product Information[57]
9	Leuprolide	Picosulfuric acid	
10	Deferasirox	Bexarotene	Ligand, Product Information[58]

Table 6. Novel predictions of drug–drug interactions. Shown are top-10 predicted drug–drug interactions together with the relevant literature providing evidence for predictions.

Discussion

We introduced SkipGNN, a novel graph neural network for predicting molecular interactions. The architecture of SkipGNN is motivated by a principle of connectivity, which we call *skip similarity*. Remarkably, we found that skip similarity allows SkipGNN to much better capture structural and evolutionary forces that govern molecular interaction networks that what is possible with current graph neural networks. SkipGNN achieves superior and robust performance on a variety of key prediction tasks in interaction networks and performs well even when networks are highly incomplete.

There are several future directions. We focused here on networks in which all edges are of the same type. As SkipGNN is a general graph neural network, it would be interesting to adapt SkipGNN to heterogeneous networks, such as drug-gene–disease networks. Another fruitful direction would be to implement skip similarity in other types of biological networks.

Appendix 1: Experiments on the importance of each layer of GNN for biomedical link prediction

To further support our claim on the importance of integrating skip similarity for GNN-based methods on biomedical interaction network link prediction, we vary the architecture of vanilla GNN and perform predictive comparison on DDI, PPI, and DTI tasks. Here are the variations:

Task	Method	PR-AUC	ROC-AUC
DTI	TwoLayers-OriGraph	0.904 ± 0.011	0.899 ± 0.010
	OneLayer-OriGraph	0.807 ± 0.024	0.781 ± 0.026
	TwoLayers-SkipGraph	0.849 ± 0.041	0.826 ± 0.052
	OneLayer-SkipGraph	0.780 ± 0.047	0.756 ± 0.046
	OneLayer-3Hops	0.806 ± 0.005	0.789 ± 0.017
DDI	TwoLayers-OriGraph	0.856 ± 0.005	0.875 ± 0.004
	OneLayer-OriGraph	0.810 ± 0.029	0.831 ± 0.029
	TwoLayers-SkipGraph	0.848 ± 0.003	0.863 ± 0.002
	OneLayer-SkipGraph	0.844 ± 0.008	0.862 ± 0.004
	OneLayer-3Hops	0.830 ± 0.013	0.851 ± 0.009
PPI	TwoLayers-OriGraph	0.909 ± 0.002	0.907 ± 0.006
	OneLayer-OriGraph	0.806 ± 0.013	0.815 ± 0.015
	TwoLayers-SkipGraph	0.900 ± 0.003	0.888 ± 0.004
	OneLayer-SkipGraph	0.873 ± 0.023	0.863 ± 0.017
	OneLayer-3Hops	0.858 ± 0.039	0.853 ± 0.022
GDI	TwoLayers-OriGraph	0.909 ± 0.002	0.906 ± 0.006
	OneLayer-OriGraph	0.846 ± 0.043	0.845 ± 0.039
	TwoLayers-SkipGraph	0.888 ± 0.008	0.905 ± 0.007
	OneLayer-SkipGraph	0.876 ± 0.016	0.883 ± 0.018
	OneLayer-3Hops	0.868 ± 0.006	0.881 ± 0.011

Table 7. *Skip Similarity* is important for biomedical interaction prediction when using GCN. Results of five independent runs on DDI, PPI and DTI tasks with varying architectures of GCN.

- **TwoLayers-OriGraph** is the two layers GCN on original graph. It uses an indirect two-hops neighborhood aggregation because the two-hops nodes information is conveyed to the center node through the one-hop nodes.
- **OneLayer-OriGraph** is a one layer vanilla GCN. It only utilizes the immediate one-hop neighbor information. Hence, it is a direct measure of *direct similarity*.
- **TwoLayers-SkipGraph** is the vanilla two layers GCN that operates on the skip graph. It uses direct connection of center node with its two-hops neighborhood as against the indirect connection in vanilla GCN. As it is two layer, it also considers indirect four-hops neighbor nodes.
- **OneLayer-SkipGraph** is the one layer version of GCN-A2. As it only uses two-hop neighbor information, it directly measures the *skip similarity*.
- **OneLayer-3Hops** is the one layer version of GCN-A3. We test to show the significance of higher order neighbors.

Table 7 compares the results. From the large improvement of TwoLayers-OriGraph over OneLayer-OriGraph, this is the initial evidence that two-hops neighborhood, which contains *skip similarity* node relation assumption, is essential. Then, comparing OneLayer-OriGraph and OneLayer-SkipGraph, the large margin improvement of OneLayer-SkipGraph implies two-hops neighbor alone has more predictive information than one-hop neighbor alone, supporting our motivation analysis of the importance of *skip similarity* for biomedical interaction network. Note also that the improvement from OneLayer-OriGraph to TwoLayers-OriGraph is much larger than the improvement from OneLayer-SkipGraph to TwoLayers-SkipGraph, meaning second-hop is essential and higher-order neighborhood is of limited importance for interaction link prediction. Lastly, TwoLayers-OriGraph performs better than TwoLayers-SkipGraph, meaning that biomedical interaction link prediction is a balance between immediate neighbor and two-hops neighbor, confirming with our intuition that an ideal network should pursue a balance between them and adding support for the iterative fusion scheme. Note that OneLayer-SkipGraph uses only the second hop neighborhood, without the first-hop neighborhood information. This suggests first-hop importance, and a necessity to integrate both first and second hop neighbors, such as SkipGNN's iterative fusion scheme. We also find 3-hops neighbor is less important than 2-hops neighbor when comparing OneLayer-SkipGraph and OneLayer-3Hops, further confirming the importance of 2-hops in biomedical interaction network.

Appendix 2: Details about baseline methods

- **L3**[15] counts the length-3 paths among all the network nodes pairs. The number of length-3 paths are then normalized by the degree of node pairs.

Task	Rank	Drug	Target Gene
DTI	1	Dpb-T	L3MBTL1
	2	Progabide	PANX1
	3	Glutamic acid	NARS2
	4	DB04530	CYP2D6
	5	Glutamic acid	AZIN2
	6	DB08152	BCHE
	7	CR002	CYP2C19
	8	Ecabet	F2
	9	Insulin	CYP2C19
	10	RU84687	CYP2C19
Task	**Rank**	**Protein 1**	**Protein 2**
PPI	1	GOLGA6A	CYSRT1
	2	CYSRT1	IPCEF1
	3	PRKAR1B	REL
	4	CEP70	UBQLN1
	5	ADAMTSL4	MTUS2
	6	SIX1	CYSRT1
	7	RBPMS	MTUS2
	8	KRT31	CCDC36
	9	TRAF2	MIPOL1
	10	DES	MEOX2
Task	**Rank**	**Gene**	**Disease**
GDI	1	RAPGEF3	Intellectual disability
	2	ISL1	Intellectual disability
	3	UHRF1BP1L	Intellectual disability
	4	RCN3	Intellectual disability
	5	UGT1A4	Schizophrenia
	6	DDX11	Schizophrenia
	7	SOD2	Mental Retardation
	8	IL1B	Diabetes mellitus type 2
	9	TNF	Osteosarcoma
	10	POMC	Tactile Allodynia

Table 8. Top-ranked novel predictions for PPI, DTI and GDI tasks. Potential novel hits for PPI, DTI, GDI tasks. For long drug names, we use the DrugBank ID instead.

- **DeepWalk**[17] performs uniform distributed random walk and applies skip-gram model to learn a node embedding. We use 20 walk lengths and then concatenate the target nodes embedding with a logistic regression classifier.
- **node2vec**[18] builds on DeepWalk and uses biased random walk based on depth/breath first search to consider both local and global network structure. We use 20 walk length as the paper suggests longer walk lengths improve the embedding quality. The paper also reported Hadamard product perform better than average and weighted L1/L2 for link prediction. However, in our experiment, the simple concatenation is better than Hadamard. After the concatenation, we feed into a logistic regression classifier as described in the paper.
- **struc2vec**[19] leverages the local network structure in addition to the node2vec. We use 80 walk length and 20 number of walks, following author's recommendation. We then concatenate the latent embedding and feed into a logistic regression classifier.
- **Spectral Clustering**[20] projects nodes on top-16 eigenvectors of the normalized Laplacian matrix and uses the transposed eigenvectors as node embeddings. The embeddings are then multiplied and pass through a sigmoid function to obtain link probabilities.
- **VGAE**[32] applies variational graph auto-encoder and learns node embeddings that best reconstruct the adjacent matrix. We use a two-layer GCN with hidden size 64 for layer one and 16 for layer two. The learning rate is set to be 5e−4 with Adam optimizer for 300 epochs. The dropout rate is set to be 0.1.
- **GCN**[9] uses two-layers GCN layers on original adjacency matrix to obtain node embeddings, others are with same setting as SkipGNN . We use a two-layer GCN with hidden size 64 for layer one and 16 for layer two. The learning rate is set to be 5e−4 with Adam optimizer for 10 epochs with batch size 256.
- **GIN**[21] uses multi-layer perceptron (MLP) as the aggregation function. We use a five layer GIN with hidden size 32. The learning rate is set to be 5e−4 with Adam optimizer for 10 epochs with batch size 256.

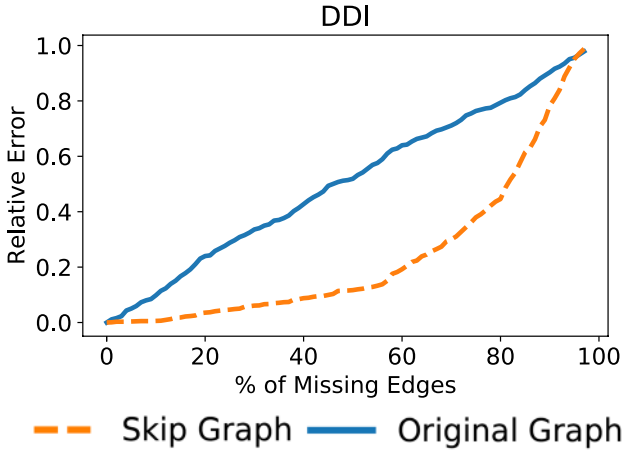

Figure 6. The ability of skip graph and original graph to capture the network structure in the face of incomplete data. Skip graph can better preserve the network structure than the original graph, as evidenced by skip graph's smaller relative error ("Robust learning on incomplete interaction network" section) than that of the original graph. This is true for all % of missing edges, indicating that skip graph can keep useful information about interaction structure even when networks are highly incomplete and many interactions are missing.

- **JK-Net**[34] uses skip connections across each layer of GNN propapagation. We use the GIN backend for JK-Net. We use three layers GIN with hidden size 64. The learning rate is set to be 5e−4 with Adam optimizer for 10 epochs with batch size 256.
- **MixHop**[11] uses multiple higher-order adjacency matrix to propagate messages. We use three layers for both the top and lower towers with size 200, 200, 200. The L2 regularization is set to be 0.0005.

We determine all parameters for the baseline methods using the random search on a validation set.

Appendix 3: Potential novel hits for PPI, DTI, and GDI

We conducted a literature search for the DDI novel hits in the main text. Here, we also provide the novel hits discovered through SkipGNN for the PPI, DTI, and GDI tasks in Table 8.

Appendix 4: A network heuristic explanation

So far, we found that SkipGNN has robust performance on incomplete interaction networks and next we investigate what makes SkipGNN to perform so robustly. We hypothesize that SkipGNN is robust because its skip graphs can preserve the graph topology much better than original graphs and this feat becomes prominent when interaction data are scarce. Note that SkipGNN uses the skip graph whereas other methods only use the original graph.

To test the hypothesis, we measure the relative error between the original graph G and the incomplete graph G^p in which edges are missing at rate p. We use a metric that calculates the relative error of the spectral norm for the graph Laplacian matrix: $\mathrm{Err}(\mathbf{A}, \mathrm{p}) = (\|\mathbf{L}\|_2 - \|\mathbf{L}^p\|_2)/\|\mathbf{L}\|_2$, where $\mathbf{L} = \mathbf{A} - \mathbf{D}$, $\mathbf{L}^p = \mathbf{A}^p - \mathbf{D}^p$, $\mathbf{A}$ ($\mathbf{A}^p$) is adjacency matrix of G (G^p), $\|\cdot\|_2 = \sigma_{\max}(\cdot)$, the $\sigma_{\max}$ is the largest singular value[60].

Figure 6 shows the relative error Err of original and skip graphs against 100 fractions p of missing edges on the DDI task. We see that the skip graph's relative error is much lower than that of original graph in almost all settings. This observation provides evidence for our hypothesis, confirming that skip graphs can better capture the graph topology than original graphs. Because of that, SkipGNN can learn high-quality embeddings even when interaction data are scarce.

Appendix 5: Biomedical interaction network visualization

A visualization of biomedical network is provided in Fig. 7.

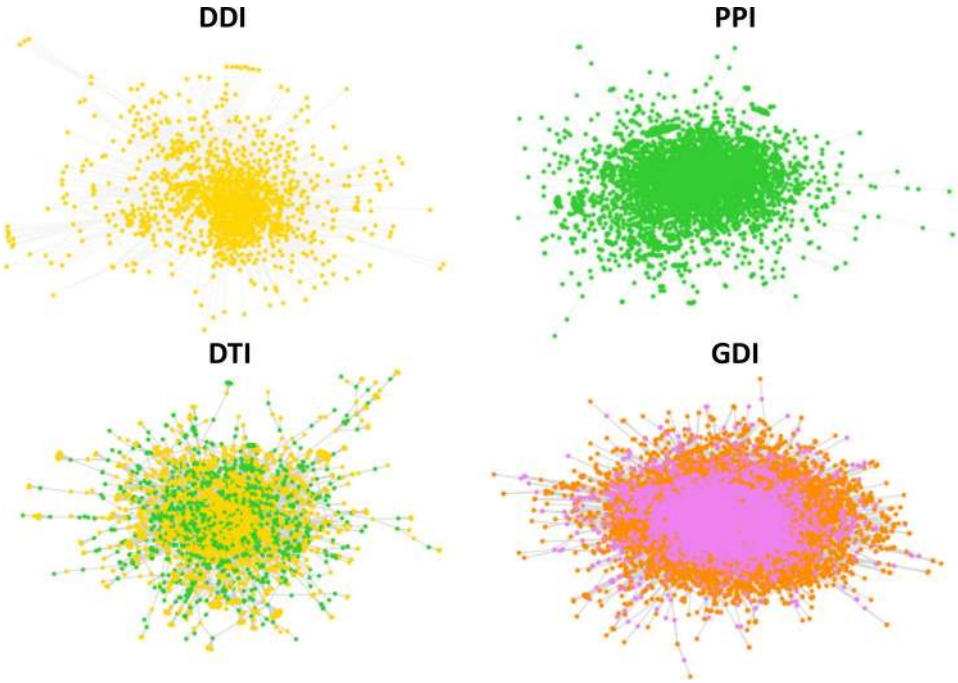

Figure 7. Network Visualization of four biomedical interaction networks.

Received: 17 June 2020; Accepted: 17 November 2020
Published online: 03 December 2020

References

1. Cowen, L., Ideker, T., Raphael, B. J. & Sharan, R. Network propagation: A universal amplifier of genetic associations. *Nat. Rev. Genet.* **18**, 551 (2017).
2. Zitnik, M. *et al.* Machine learning for integrating data in biology and medicine: Principles, practice, and opportunities. *Inf. Fusion* **50**, 71–91 (2019).
3. Luo, Y. *et al.* A network integration approach for drug–target interaction prediction and computational drug repositioning from heterogeneous information. *Nat. Commun.* **8**, 1–13 (2017).
4. Zitnik, M., Agrawal, M. & Leskovec, J. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics* **34**, i457–i466 (2018).
5. Luck, K. *et al.* A reference map of the human binary protein interactome. *Nature* **580**, 402–408 (2020).
6. Agrawal, M., Zitnik, M. & Leskovec, J. Large-scale analysis of disease pathways in the human interactome. In *PSB* 111–122 (2018).
7. Lei, C. & Ruan, J. A novel link prediction algorithm for reconstructing protein–protein interaction networks by topological similarity. *Bioinformatics* **29**, 355–364 (2013).
8. Wu, Z. *et al.* A comprehensive survey on graph neural networks. arXiv:1901.00596 (2019).
9. Kipf, T. N. & Welling, M. Semi-supervised classification with graph convolutional networks. In *ICLR* (2017).
10. Veličković, P. *et al.* Graph attention networks. In *ICLR* (2018).
11. Abu-El-Haija, S. *et al.* Mixhop: Higher-order graph convolution architectures via sparsified neighborhood mixing. In *ICML* (2019).
12. Costanzo, M. *et al.* The genetic landscape of a cell. *Science* **327**, 425–431 (2010).
13. Costanzo, M. *et al.* A global genetic interaction network maps a wiring diagram of cellular function. *Science* **353**, aaf1420 (2016).
14. Zitnik, M. *et al.* Evolution of resilience in protein interactomes across the tree of life. *PNAS* **116**, 4426–4433 (2019).
15. Kovács, I. A. *et al.* Network-based prediction of protein interactions. *Nat. Commun.* **10**, 1240 (2019).
16. McPherson, M., Smith-Lovin, L. & Cook, J. M. Birds of a feather: Homophily in social networks. *Ann. Rev. Sociol.* **27**, 415–444 (2001).
17. Perozzi, B., Al-Rfou, R. & Skiena, S. DeepWalk: Online learning of social representations. In *KDD* 701–710 (2014).
18. Grover, A. & Leskovec, J. node2vec: Scalable feature learning for networks. In *KDD* 855–864 (2016).
19. Ribeiro, L. F., Saverese, P. H. & Figueiredo, D. R. struc2vec: Learning node representations from structural identity. In *KDD* 385–394 (2017).
20. Tang, L. & Liu, H. Leveraging social media networks for classification. *Data Min. Knowl. Disc.* **23**, 447–478 (2011).
21. Xu, K., Hu, W., Leskovec, J. & Jegelka, S. How powerful are graph neural networks? In *ICLR* (2018).
22. Lü, L. & Zhou, T. Link prediction in complex networks: A survey. *Physica A* **390**, 1150–1170 (2011).
23. Menche, J. *et al.* Uncovering disease–disease relationships through the incomplete interactome. *Science* **347**, 1257601 (2015).
24. Durán, C. *et al.* Pioneering topological methods for network-based drug–target prediction by exploiting a brain-network self-organization theory. *Brief. Bioinform.* **19**, 1183–1202 (2018).
25. Barabási, A.-L. & Albert, R. Emergence of scaling in random networks. *Science* **286**, 509–512 (1999).
26. Lü, L., Jin, C.-H. & Zhou, T. Similarity index based on local paths for link prediction of complex networks. *Phys. Rev. E* **80**, 046122 (2009).
27. Zitnik, M. & Zupan, B. Data fusion by matrix factorization. *IEEE Trans. Pattern Anal. Mach. Intell.* **37**, 41–53 (2015).
28. Wang, B. *et al.* Network enhancement as a general method to denoise weighted biological networks. *Nat. Commun.* **9**, 1–8 (2018).
29. Xu, L., Cao, J., Wei, X. & Yu, P. Network embedding via coupled kernelized multi-dimensional array factorization. *IEEE TKDE* (2019).
30. Tang, J. *et al.* Line: Large-scale information network embedding. In *WWW* 1067–1077 (2015).

31. Hamilton, W., Ying, Z. & Leskovec, J. Inductive representation learning on large graphs. In *NeurIPS* 1024–1034 (2017).
32. Kipf, T. N. & Welling, M. Variational graph auto-encoders. In *NeuralIPS Workshop on Bayesian Deep Learning* (2016).
33. Ma, T., Xiao, C., Zhou, J. & Wang, F. Drug similarity integration through attentive multi-view graph auto-encoders. In *IJCAI* (2018).
34. Xu, K. *et al.* Representation learning on graphs with jumping knowledge networks. In *ICML* (2018).
35. Tsubaki, M., Tomii, K. & Sese, J. Compound-protein interaction prediction with end-to-end learning of neural networks for graphs and sequences. *Bioinformatics* **35**, 309–318 (2019).
36. Öztürk, H., Özgür, A. & Ozkirimli, E. Deepdta: deep drug–target binding affinity prediction. *Bioinformatics* **34**, i821–i829 (2018).
37. Gao, Y. *et al.* Interpretable drug target prediction using deep neural representation. In *IJCAI* 3371–3377 (2018).
38. Huang, K., Xiao, C., Hoang, T. N., Glass, L. M. & Sun, J. Caster: Predicting drug interactions with chemical substructure representation. In *AAAI* (2020).
39. Ryu, J. Y., Kim, H. U. & Lee, S. Y. Deep learning improves prediction of drug–drug and drug–food interactions. *PNAS* **115**, E4304–E4311 (2018).
40. Cheng, F. & Zhao, Z. Machine learning-based prediction of drug–drug interactions by integrating drug phenotypic, therapeutic, chemical, and genomic properties. *JAMIA* **21**, e278–e286 (2014).
41. Milenković, T. & Pržulj, N. Uncovering biological network function via graphlet degree signatures. *Cancer Inform.***6**, CIN–S680 (2008).
42. Zhang, W. *et al.* Predicting drug-disease associations by using similarity constrained matrix factorization. *BMC Bioinform.* **19**, 1–12 (2018).
43. Ferdousi, R., Safdari, R. & Omidi, Y. Computational prediction of drug–drug interactions based on drugs functional similarities. *JBI* **70**, 54–64 (2017).
44. Zhang, P., Wang, F., Hu, J. & Sorrentino, R. Label propagation prediction of drug–drug interactions based on clinical side effects. *Sci. Rep.* **5**, 1–10 (2015).
45. Zitnik, M. & Leskovec, J. Predicting multicellular function through multi-layer tissue networks. *Bioinformatics* **33**, i190–i198 (2017).
46. Cao, W., Yan, Z., He, Z. & He, Z. A comprehensive survey on geometric deep learning. *IEEE Access* **8**, 35929–35949 (2020).
47. Zitnik, M., Sosič, R., Maheshwari, S. & Leskovec, J. BioSNAP Datasets: Stanford biomedical network dataset collection (2018).
48. Luck, K. *et al.* A reference map of the human protein interactome. *bioRxiv* (2019).
49. Piñero, J. *et al.* The DisGeNET knowledge platform for disease genomics: 2019 update. *Nucleic Acids Res.* **48**, 845–855 (2019).
50. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. In *ICLR* (2014).
51. Zhang, M. & Chen, Y. Link prediction based on graph neural networks. In *NeurIPS* 5165–5175, (2018).
52. Maaten, L. v. d. & Hinton, G. Visualizing data using t-SNE. *JMLR* **9**, 2579–2605 (2008).
53. Mukku, S. S. R., Sivakumar, P. & Varghese, M. Clozapine use in geriatric patients–challenges. *Asian J. Psychiatry* **33**, 63–67 (2018).
54. Robertson, S. M. *et al.* Clinical drug–drug interaction assessment of ivacaftor as a potential inhibitor of cytochrome p450 and p-glycoprotein. *J. Clin. Pharmacol.* **55**, 56–62 (2015).
55. DuPont, P. *Product Information. Coumadin (Warfarin)*. (DuPont Pharmaceuticals, Wilmington, 2000).
56. Snyder, D. S. Interaction between Cyclosporine and Warfarin. *Ann. Intern. Med.* **108**, 311 (1988).
57. Merck, C. I. *Product Information. Belsomra (Suvorexant)*. (Merck & Company Inc., Whitehouse Station, 2014).
58. Ligand, P. *Product Information. Targretin (Bexarotene)*. (Ligand Pharmaceuticals, San Diego, 1999).
59. Novartis, P. Product Information. Clozaril (Clozapine). (Novartis Pharmaceuticals, East Hanover, 1989).
60. Chung, F. R. & Graham, F. C. *Spectral Graph Theory*. Vol. 92 (American Mathematical Soc., Providence, 1997).

Author contributions

K.H., C.X., J.S. conceived the projects, K.H., C.X., M.Z., J.S. conceived the experiments, K.H. conducted the experiments. All authors analyzed the results and reviewed the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to J.S.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.