SkipGNN: Predicting Molecular Interactions with Skip-Graph Networks

Kexin Huang¹, Cao Xiao², Lucas M. Glass², Marinka Zitnik³, and Jimeng Sun⁴

¹Health Data Science, Harvard T.H. Chan School of Public Health, Boston, MA
²Analytic Center of Excellence, IQVIA, Cambridge, MA
³Department of Biomedical Informatics, Harvard University, Boston, MA
⁴Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL

Abstract

Motivation: Molecular interaction networks are powerful resources for the 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 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.

Results: 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, outperforming existing methods by up to 28.8% of area under the precision recall curve (PR-AUC). Furthermore, we show that unlike popular GNNs, SkipGNN learns biologically meaningful embeddings and performs especially well on noisy, incomplete interaction networks.

Availability: SkipGNN method, experiments, and all datasets will be open-sourced through https://github.com/kexinhuang12345/SkipGNN.

Contact: jimeng@illinois.edu

1 Introduction

Molecular interaction networks are ubiquitous in biological systems. Over the last decade, interaction networks have advanced our systems-level understanding of biology [Cowen et al., 2017]. Further, they have enabled discovery of biologically significant, yet previously unmapped relationships [Zitnik et al., 2019a],
Direct similarity

Skip similarity

**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 [McPherson et al., 2001]. (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 [Costanzo et al., 2010, 2016] and protein-protein interaction networks [Kovács et al., 2019, Zitnik et al., 2019b].

including drug-target interactions (DTIs) [Luo et al., 2017a], drug-drug interactions (DDIs) [Zitnik et al., 2018a], protein-protein interactions (PPIs) [Luck et al., 2020], and disease-gene associations (DGIs) [Agrawal et al., 2018]. To assist in these discoveries, a plethora of computational methods, primarily optimized for link prediction from networks (e.g., Lei and Ruan [2013]), 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) [Wu et al., 2019] and graph convolutional networks (GCNs) [Kipf and Welling, 2017, Veličković et al., 2018], 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 [Abu-El-Haija et al., 2019].

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 [Costanzo et al., 2010, 2016, Zitnik et al., 2019b, Kovács et al., 2019]. This is because interactions can exist between nodes that are not necessarily similar, as illustrated in Figure 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 to as skip similarity (Figure 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 [Kipf and Welling, 2017], 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 (Figure 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, and network metrics by up to 28.8% in PR-AUC [Perozzi et al., 2014, Grover and Leskovec, 2016, Ribeiro et al., 2017, Tang and Liu, 2011, Kovács et al., 2019]. Further, the method shows a 7.9% improvement in PR-AUC over state-of-the-art graph neural networks [Kipf and Welling, 2017, 2016].

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 (Section 5.1-5.2). 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 (Section 5.3), which is also confirmed experimentally (Section 5.5).

Further, we examine embeddings learned by SkipGNN and find that SkipGNN learns biologically meaningful embeddings, whereas a regular GCN does not (Section 5.4). 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 (Section 5.6).

2 Related work

Related work on link prediction. Existing link prediction methods belong to one of the following categories. (1) Heuristic or mechanistic methods (e.g., Lü and Zhou [2011], Menche et al. [2015], Durán et al. [2018], Kovács et al. [2019]) 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) [Barabási and Albert, 1999] and Local Path Index (LP) [Lü et al., 2009]. However, these methods usually make strong assumptions about the network structure and hence suffer from instability of performance [Lü and Zhou, 2011, Kovács et al., 2019]. (2) Direct embedding methods generate embeddings for every node in the network capturing the node’s local network topology (e.g., Zitnik and Zupan [2015], Wang et al. [2018], Xu et al. [2019]). A popular approach is to use random walks with a skip-gram model, such as DeepWalk [Perozzi et al., 2014], node2vec [Grover and Leskovec, 2016], and LINE [Tang et al., 2015]. The other popular approach leverages the spectral graph theory to generate a spectral embedding such as spectral clustering [Tang and Liu, 2011]. 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) [Kipf and Welling, 2017, Hamilton et al., 2017], Variational Graph Autoencoders (VGAE) [Kipf and Welling, 2016, Ma et al., 2018], and Graph Attention Networks (GAT) [Veličković et al., 2018] 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). GNN-based methods have been shown to achieve state-of-the-art performance across a variety of network datasets [Zhang and Chen, 2018, Kipf and Welling, 2016]. SkipGNN further improves GNN-based methods by capturing the skip similarity that other GNN-based methods are incapable of capturing.

Related work on molecular interaction prediction. 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, Tsubaki et al. [2019], Öztürk et al. [2018], Gao et al. [2018] use graph-convolutional (GCN) and convolutional (CNN) networks on molecular graphs and gene sequence data to predict binding of drugs to target proteins. Similarly, Huang et al. [2020], Ryu et al. [2018], Cheng and Zhao [2014] 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 [Milenković and Pržulj, 2008]) 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 [Zhang et al., 2018], clustering [Ferdousi et al., 2017], and label propagation [Zhang et al., 2015]. (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 [Zitnik et al., 2018a, Ma et al., 2018, Zitnik and Leskovec, 2017]. SkipGNN belongs to the structural representation learning category.

**Background on Graph Neural Networks (GNNs).** Next, we describe graph neural networks as they are 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 $A$. 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., Kipf and Welling [2017]):

$$H^{(l+1)} = \sigma(D^{-\frac{1}{2}}AD^{-\frac{1}{2}}H^{(l)}W),$$

where $H^{(0)}$ is a matrix of node embeddings at the $l$-th layer, $H^{(0)}$ are input node attributes, $W$ is a trainable parameter matrix, $\sigma$ is a non-linear activation function, and $D$ and $\tilde{A}$ are the renormalized degree and adjacency matrices, defined as: $\tilde{A} = A + I$ and $D = \sum_j \tilde{A}_{ij}$ ($I$ 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. However, GNN is limited at capturing skip similarity, where SkipGNN utilizes an additional skip-graph to fully exploit this important quality for biomedical interaction network.

**3 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.

**Problem formulation.** Consider an interaction network $G$ on $N$ nodes representing biomedical entities $V$ (e.g., drugs, proteins, or diseases) and $M$ edges $E$ 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 [Luo et al., 2017b]. 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 $A$, where $A_{ij}$ is $1$ if nodes $i$ and $j$ are connected ($e_{ij} = 1$) in the graph and otherwise $0$ ($e_{ij} = 0$). Further, $D$ is the degree matrix, a diagonal matrix, where $D_{ii}$ is the degree of node $i$.

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

**3.1 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
Neural architecture of SkipGNN. (Left) SkipGNN constructs skip graph $G_s$ (denoted by adjacency matrix $A_s$) based on the input graph $G$ (denoted by adjacency matrix $A$) using Eq. (i). (Middle) Initial node embeddings, $H^{(0)}$ and $S^{(0)}$, are specified using side information (e.g., gene expression vectors if nodes represent genes) or generated using node2vec [Grover and Leskovec, 2016]. 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 $W_s^{(0)}$ and $W_o^{(0)}$ (operating on $A$ and $A_s$, respectively) are fused via weight matrices $W_s^{(0)}$ and $W_o^{(0)}$ based on Eq. (2, 3). This completes computations in the first layer of SkipGNN, producing embeddings $H^{(1)}$ and $S^{(1)}$. In the second layer, those embeddings are transformed via $W_s^{(1)}$ and $W_o^{(1)}$ using Eq. (4), resulting in final embeddings $E$. (Right) Embeddings $E_i$ and $E_j$ of target nodes $i$ and $j$ are retrieved, concatenated, and then fed into a decoder (parameterized by $W_d$). Decoder returns $p_{ij}$, representing the probability that nodes $i$ and $j$ interact.

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 $A_s$:

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

The corresponding degree matrix is $D_s^{ij} = \sum_j A_s^{ij}$. An efficient way to implement the skip graph is through matrix multiplication:

$$
A_s = \text{sign}(A A^T),
$$

(1)

where: $\text{sign}(x)$ is the sign function, $\text{sign}(x) = 1$ if $x > 0$ and 0 otherwise, which is applied element-wise on $AA^T$. It counts the number of two-hop paths from node $i$ to $j$. Hence, if an entry for node $i, j$ in $AA^T$ is larger than 0, it means there exists a skipped node between node $i, j$. Then, we convert the positive entry into 1 to
Table 1: Notation used in SkipGNN.

| Notation | Definition |
|----------|------------|
| \( G : \{\mathcal{V}, \mathcal{E}\} \) | Graph with nodes \( \mathcal{V} \) and edges \( \mathcal{E} \) |
| \( \mathbf{D}, \mathbf{A} \in \mathbb{Z}^{N \times N} \) | Degree and adjacency matrices for graph \( G \) |
| \( \tilde{\mathbf{D}}, \tilde{\mathbf{A}} \in \mathbb{Z}^{N \times N} \) | 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 |
| \( \mathbf{D}_s, \mathbf{A}_s \in \mathbb{Z}^{N \times N} \) | Degree and adjacency matrices for \( G_s \) |
| \( \tilde{\mathbf{D}}_s, \tilde{\mathbf{A}}_s \in \mathbb{R}^{N \times N} \) | Normalized degree and adjacency matrices for \( G_s \) |
| \( H^{(l)}, S^{(l)} \) | Node embeddings for \( G \) and \( G_s \), in layer \( l \) |
| \( E \) | 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 \( j \) |
| \( \mathcal{L} \in \mathbb{R} \) | Binary classification loss |
| \( W_o^{(l)}, W_s^{(l)} \) | Weight matrix for original (o) and skip (s) graphs, layer \( l \) |
| \( W_o^{(l)} \) | Weight matrix for skip-to-original-graph fusion |
| \( W_s^{(l)} \) | Weight matrix for original-to-skip-graph fusion |
| \( W_d, b \) | Decoder weight matrix and bias parameter |

construct the skip graph’s adjacent matrix. Given this skip graph, we proceed to describe the full SkipGNN model.

3.2 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 Section 3.1, 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.

3.2.1 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 Figure 3):

\[
H^{(l+1)} = \sigma(AGG(FH^{(l)}W_o^{(l)} + F_sS^{(l)}W_o^{(l)})) \\
S^{(l+1)} = \sigma(AGG(F_sS^{(l)}W_s^{(l)} + FH^{(l+1)}W_s^{(l)})),
\]

where:

\[
F = \tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{A}\tilde{\mathbf{D}}^{-\frac{1}{2}}, \quad F_s = \tilde{\mathbf{D}}_s^{-\frac{1}{2}}\mathbf{A}_s\tilde{\mathbf{D}}_s^{-\frac{1}{2}}.
\]
Here, \( H^{(l)}, S^{(l)} \) is node embeddings at the \( l \)-th layer from direct similarity graph \( G \) and skip similarity graph \( G_s \), respectively. \( F, F_i \) are the re-normalized adjacency matrices from direct similarity and skip similarity, respectively. And \( W_o^{(l)}, W'_o^{(l)}, W_s^{(l)}, W'_s^{(l)} \) are the transformed weights for layer \( l \). \( H^{(0)} \) and \( S^{(0)} \) are set to be \( X \), the input node attributes generated from node2vec. The aggregate gate \( AGG \) in Eq. (2-3) can be a summation, a Hadamard product, max-pooling, or some other aggregation operator [Cao et al., 2020]. Empirically, we find that summation gate has the best performance. \( \sigma() \) is the activation function and we use ReLU() = \( \max(\cdot, 0) \) to add non-linearity in the propagation.

In each iteration, the node embedding for original graph \( H^{(l+1)} \) is first updated with its previous layer’s node embedding \( H^{(l)} \), combined with skip graph embedding \( S^{(l)} \). After obtaining the updated original graph embedding \( H^{(l+1)} \), we then update the skip graph embedding \( S^{(l+1)} \) 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 \( E \) is obtained through:

\[
E = AGG(FH^{(L_{\text{max}})}W_o^{(L_{\text{max}})}, FS^{(L_{\text{max}})}W_s^{(L_{\text{max}})}),
\]

where \( L_{\text{max}} \) is the last layer index. As in the motivation, we are interested only in up to second order neighbors, we set \( L_{\text{max}} = 1 \), see Figure 3. 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 Section 5.5.

### 3.2.2 SkipGNN decoder

Given the target nodes \((i, j)\) and their corresponding node embedding \( E_i, E_j \), we implement a neural network as a decoder to first combine \( E_i, E_j \) 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 \( W_d \) and bias \( b \) as a binary classifier to obtain probability \( p_{ij} \):

\[
p_{ij} = \sigma(W_d \text{COMB}(E_i, E_j) + b),
\]

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.

### 3.3 The SkipGNN algorithm

The overall algorithm is shown in Algorithm 1. Here, we only leverage accessible network information (adjacent matrix \( A \) of the network \( G \)) to predict links. In all experiments, we initialize embeddings using node2vec [Grover and Leskovec, 2016] as: \( X = \text{node2vec}(A) \).

Second, we construct the skip graph with adjacent matrix \( A_s \) via Eq. (1) to capture the skip-similarity principle. Next, at every step, a mini-batch of interaction pairs \( \mathcal{M} \) 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), Eq. (3)) 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 \( E \). We then retrieve the embeddings for each node in the mini-batched pairs \( \mathcal{M} \) and concatenate them to feed into decoder (Eq. (5)).

During training, we optimize the SkipGNN’s parameters \( W_o^{(l)}, W'_o^{(l)}, W_s^{(l)}, W'_s^{(l)}, W_d, b \) in an end-to-end manner through a binary classification loss: \( \mathcal{L} = \sum_{(i,j) \in \mathcal{M}} y_{ij} \log p_{ij} + (1 - y_{ij}) \log (1 - p_{ij}) \), where \( y_{ij} \) is the true label for nodes \( i \) and \( j \) that are sampled during training via mini-batching, \((i, j) \in \mathcal{M} \), and \( \mathcal{M} \) 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.
Algorithm 1: The SkipGNN Algorithm

**Input:** interaction network $G$ with adjacent matrix $A$

**Node Embedding Generation, e.g.:**

$X \leftarrow \text{node2vec}(A)$

**Skip Graph Construction (Section 3.1):**

$A_s \leftarrow \text{SkipGraph}(A)$ via Eq. (1)

for $t = 1 \ldots T_{\text{max}}$ do

sample mini-batch of training node pairs $\mathcal{M} \subseteq \mathcal{E}$ with corresponding labels $y$

for $l = 0 \ldots (L_{\text{max}} - 1)$ do

**Iterative Fusion (Section 3.2.1):**

$H^{(l+1)} \leftarrow \text{FusionGate}(H^{(l)})$ via Eq. (2)

$S^{(l+1)} \leftarrow \text{FusionGate}(S^{(l)})$ via Eq. (3)

$E \leftarrow \text{FuseGate}(H^{(L_{\text{max}})}, S^{(L_{\text{max}})})$ via Eq. (4)

**Decoder (Section 3.2.2):**

$p_{ij} \leftarrow \text{decode}(E)$ via Eq. (5)

Compute the loss value $\mathcal{L}$ using $p_{ij}$ and $y$ (Sec. 3.3) and update model parameters via gradient descent

---

4 Data and experimental setup

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

4.1 Molecular interaction networks

We consider four publicly-available network datasets. (1) BIOSNAP-DTI [Zitnik et al., 2018b] contains 5,018 drugs that target 2,325 protein through 15,139 drug-target (DTI) interactions. (2) BIOSNAP-DDI [Zitnik et al., 2018b] consists of 48,514 drug-drug interactions (DDIs) between 1,514 drugs extracted from drug labels and biomedical literature. (3) HuRI-PPI [Luck et al., 2019] 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 [Piñero et al., 2019] collects curated disease-gene associations (GDIs) from GWAS studies, animal models and scientific literature. The dataset has 81,746 interactions between 9,413 genes and 10,370 diseases.

4.2 SkipGNN implementation and hyperparameters

We implemented SkipGNN using PyTorch deep learning framework\(^1\). We use a server with 2 Intel XeonE-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 [Kingma and Ba, 2014], mini-batch size is $|\mathcal{M}| = 256$, epoch size is 15, and dropout rate is 0.1. We set hyper-parameters using grid search based on best average prediction performance on validation set. Specifically, we use $L_{\text{max}} = 1$, hidden size in the first layer as $d^{(1)} = 64$ and hidden size in the second layer as $d^{(2)} = 16$.

4.3 Baseline methods

We compare SkipGNN to seven powerful predictors of molecular interactions from network science and graph machine-learning fields.

---

\(^1\)The source code implementation of SkipGNN is available at https://github.com/kexinhuang12345/SkipGNN.
Table 2: **Data statistics.** 'A' indicates average node degree.

| Dataset    | Prediction task               | # nodes | # edges       | A    |
|------------|-------------------------------|---------|---------------|------|
| DTI        | Drug-target interaction       | 7,343   | 15,139        | 4.12 |
| DDI        | Drug-drug interaction         | 1,514   | 48,514        | 64.09|
| PPI        | Protein-protein interaction   | 5,604   | 23,322        | 8.32 |
| GDI        | Disease-gene interaction      | 19,783  | 81,746        | 8.26 |

From machine learning, we use three direct network embedding methods: **DeepWalk** [Perozzi et al., 2014], **node2vec** [Grover and Leskovec, 2016], and we also include **struc2vec** [Ribeiro et al., 2017]. 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 two graph neural networks: **VGAE** [Kipf and Welling, 2016] and **GCN** [Kipf and Welling, 2017]. From network science, we consider **Spectral Clustering** [Tang and Liu, 2011]. We also use **L3** [Kovács et al., 2019] 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.

### 4.4 Experimental setup

In all our experiments, we follow an established evaluation strategy for link prediction (e.g., Zhang and Chen [2018], Zitnik et al. [2018a]). We divide each dataset into train, validation, and test sets in a 7:1:2 ratio, which yields positive examples (molecular interactions). We generate their negative counterparts by sampling the complement set of positive examples. 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: (i) area under precision-recall curve (PR-AUC): 
\[
\text{PR-AUC} = \sum_{k=1}^{n} \text{Prec}(k) \cdot \Delta \text{Rec}(k),
\]
where \(k\) is \(k\)-th precision/recall operating point (\(\text{Prec}(k)\), \(\text{Rec}(k)\)); and (2) area under the receiver operating characteristics curve (ROC-AUC): 
\[
\text{ROC-AUC} = \sum_{k=1}^{n} \text{TP}(k) \cdot \Delta \text{FP}(k),
\]
where \(k\) is \(k\)-th true-positive and false-positive operating point (\(\text{TP}(k)\), \(\text{FP}(k)\)). Higher values of PR-AUC and ROC-AUC indicate better predictive performance.

### 5 Results

Next, we conduct a variety of experiments to investigate the predictive power of SkipGNN (Section 5.1). We then study the method’s robustness to noise and missing data (Section 5.2 and Section 5.3) and demonstrate the **skip similarity** principle (Section 5.4). Next, we conduct ablation studies to examine contributions of each of SkipGNN’s components towards the final SkipGNN performance (Section 5.5). Finally, we investigate novel predictions made by SkipGNN (Section 5.6).

#### 5.1 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 associations, 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.
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 and DTI tasks on state of the art link prediction algorithms.

| Task | Method    | PR-AUC    | ROC-AUC   |
|------|-----------|-----------|-----------|
| DTI  | DeepWalk  | 0.753 ± 0.008 | 0.735 ± 0.009 |
|      | node2vec  | 0.771 ± 0.005 | 0.720 ± 0.010 |
|      | struc2vec | 0.677 ± 0.007 | 0.656 ± 0.010 |
|      | SC        | 0.818 ± 0.007 | 0.743 ± 0.008 |
|      | L3        | 0.891 ± 0.004 | 0.793 ± 0.006 |
|      | VGAE      | 0.853 ± 0.010 | 0.800 ± 0.010 |
|      | GCN       | 0.904 ± 0.011 | 0.899 ± 0.010 |
|      | SkipGNN   | **0.928 ± 0.006** | **0.922 ± 0.004** |
| DDI  | DeepWalk  | 0.698 ± 0.012 | 0.712 ± 0.009 |
|      | node2vec  | 0.801 ± 0.004 | 0.809 ± 0.002 |
|      | struc2vec | 0.643 ± 0.012 | 0.654 ± 0.007 |
|      | SC        | 0.749 ± 0.009 | 0.816 ± 0.006 |
|      | L3        | 0.860 ± 0.004 | 0.869 ± 0.003 |
|      | VGAE      | 0.844 ± 0.076 | 0.878 ± 0.008 |
|      | GCN       | 0.856 ± 0.005 | 0.875 ± 0.004 |
|      | SkipGNN   | **0.866 ± 0.006** | **0.886 ± 0.003** |
| PPI  | DeepWalk  | 0.715 ± 0.008 | 0.706 ± 0.005 |
|      | node2vec  | 0.773 ± 0.010 | 0.766 ± 0.005 |
|      | struc2vec | 0.875 ± 0.004 | 0.868 ± 0.006 |
|      | SC        | 0.897 ± 0.003 | 0.859 ± 0.003 |
|      | L3        | 0.899 ± 0.003 | 0.861 ± 0.003 |
|      | VGAE      | 0.875 ± 0.004 | 0.844 ± 0.006 |
|      | GCN       | 0.909 ± 0.002 | 0.907 ± 0.006 |
|      | SkipGNN   | **0.921 ± 0.003** | **0.917 ± 0.004** |
| GDI  | DeepWalk  | 0.827 ± 0.007 | 0.832 ± 0.003 |
|      | node2vec  | 0.828 ± 0.006 | 0.834 ± 0.003 |
|      | struc2vec | 0.910 ± 0.006 | 0.909 ± 0.005 |
|      | SC        | 0.905 ± 0.002 | 0.863 ± 0.003 |
|      | L3        | 0.899 ± 0.001 | 0.832 ± 0.001 |
|      | VGAE      | 0.902 ± 0.006 | 0.873 ± 0.009 |
|      | GCN       | 0.909 ± 0.002 | 0.906 ± 0.006 |
|      | SkipGNN   | **0.915 ± 0.003** | **0.912 ± 0.004** |
Figure 3: Predictive performance as a function of network incompleteness. SkipGNN provides robust result in varying fraction of missing edges. Five-fold 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. SKIP-GCN consistently shows the best performance even when networks are highly incomplete.

We report results in Table 4.3. We see that SkipGNN has outperforms all baseline methods across all molecular interaction networks. Specifically, we see up to 2.7% improvement of SkipGNN over GCN and up to 8.8% improvement over VGAE on PR-AUC. While 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 [Zitnik et al., 2018a] 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.

5.2 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 Figure 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,
Figure 4: 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 (Section 5.2) 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.

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 node/two.vec 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.

5.3 Skip graph better captures noisy molecular interactions

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: 

\[
\text{Err}(A, p) = \frac{\|L\|_2 - \|L^p\|_2}{\|L\|_2},
\]

where \(L = A - D\), \(L^p = A^p - D^p\), \(A\) (\(A^p\)) is adjacency matrix of \(G\) (\(G^p\)), \(\| \cdot \|_2 = \sigma_{\max}(\cdot)\), the \(\sigma_{\max}\) is the largest singular value [Chung and Graham, 1997].

Figure 4 shows the relative error \(\text{Err}\) 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.

5.4 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 [Maaten and Hinton, 2008]
Figure 5: **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.

Figure 6: **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.
Table 4: **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.

| 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 |

and visualize the learned embeddings in Figure 5 (DTI network) and Figure 6 (GDI network).

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).

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*.

### 5.5 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 method performance changes when we remove each of the components:
Table 5: **Novel predictions of drug-drug interactions.** Shown are top-10 predicted drug-drug interactions together with the relevant literature providing evidence for predictions.

| Rank | Drug 1   | Drug 2       | Evidence for DDI       |
|------|----------|--------------|------------------------|
| 1    | Warfarin | Clozapine    | Mukku et al. [2018]    |
| 2    | Warfarin | Ivacaftor    | Robertson et al. [2015]|
| 3    | Phenelzine | Deferasirox |                        |
| 4    | Warfarin | Paraldehyde  | DuPont [2000]          |
| 5    | Warfarin | Cyclosporine | Snyder [1988]          |
| 6    | Phenytoin | Sipuleucel-T |                        |
| 7    | Warfarin | Netupitant   |                        |
| 8    | Phenelzine | Suvorexant  | Merck [2014]           |
| 9    | Leuprolide | Picosulfuric acid |             |
| 10   | Deferasirox | Bexarotene | Ligand [1999]          |

- **-fusion** removes SkipGNN’s fusion scheme and replaces it 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 \( \text{AGG}(A, B) = |A - B| \), where \(| \cdot |\) is the absolute value operator.
- **-Hadamard** replaces the summation gate with Hadamard operator ‘\(*\)’ in Eq. (2) such that \( \text{AGG}(A, B) = A * B \).

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.

### 5.6 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 5.6). 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, Mukku et al. [2018] 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 Novartis [1989] 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, Robertson et al. [2015] 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 C.

6 Conclusion

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 than 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.

References

Lenore Cowen, Trey Ideker, Benjamin J Raphael, and Roded Sharan. Network propagation: a universal amplifier of genetic associations. *Nature Reviews Genetics*, 18(9):551, 2017.

Marinka Zitnik, Francis Nguyen, Bo Wang, Jure Leskovec, Anna Goldenberg, and Michael M Hoffman. Machine learning for integrating data in biology and medicine: Principles, practice, and opportunities. *Information Fusion*, 50:71–91, 2019a.

Yunan Luo et al. A network integration approach for drug-target interaction prediction and computational drug repositioning from heterogeneous information. *Nature Communications*, 2017a.

Marinka Zitnik, Monica Agrawal, and Jure Leskovec. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*, 34(15):i457–i466, 2018a.

Katja Luck et al. A reference map of the human binary protein interactome. *Nature*, pages 1–7, 2020.

Monica Agrawal, Marinka Zitnik, and Jure Leskovec. Large-scale analysis of disease pathways in the human interactome. In *PSB*, pages 111–122, 2018.

Chengwei Lei and Jianhua Ruan. A novel link prediction algorithm for reconstructing protein–protein interaction networks by topological similarity. *Bioinformatics*, 29(3):355–364, 2013.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596*, 2019.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *ICLR*, 2018.

Sami Abu-El-Haija et al. Mixhop: Higher-order graph convolution architectures via sparsified neighborhood mixing. *ICML*, 2019.
Michael Costanzo, Anastasia Baryshnikova, Jeremy Bellay, Yungil Kim, Eric D Spear, Carolyn S Sevier, Huiming Ding, Judice LY Koh, Kiana Toufighi, Sara Mostafavi, et al. The genetic landscape of a cell. Science, 327(5964):425–431, 2010.

Michael Costanzo, Benjamin VanderSluis, Elizabeth N Koch, Anastasia Baryshnikova, Carles Pons, Guihong Tan, Wen Wang, Matej Usaj, Julia Hanchard, Susan D Lee, et al. A global genetic interaction network maps a wiring diagram of cellular function. Science, 355(6306):aaaf4220, 2016.

Marinka Zitnik, Marcus W Feldman, Jure Leskovec, et al. Evolution of resilience in protein interactomes across the tree of life. PNAS, 116(10):4426–4433, 2019b.

István A Kovács et al. Network-based prediction of protein interactions. Nature Communications, 10(1):1240, 2019.

Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. Annual Review of Sociology, 27(1):415–444, 2001.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. DeepWalk: Online learning of social representations. In KDD, pages 701–710. ACM, 2014.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In KDD, pages 855–864, 2016.

Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. struc2vec: Learning node representations from structural identity. In KDD, pages 385–394. ACM, 2017.

Lei Tang and Huan Liu. Leveraging social media networks for classification. Data Mining and Knowledge Discovery, 23(3):447–478, 2011.

Thomas N Kipf and Max Welling. Variational graph auto-encoders. NeurIPS Workshop on Bayesian Deep Learning, 2016.

Linyuan Lü and Tao Zhou. Link prediction in complex networks: A survey. Physica A: Statistical Mechanics and its Applications, 390(6):1150–1170, 2011.

Jörg Menche et al. Uncovering disease-disease relationships through the incomplete interactome. Science, 347(6224):1257601, 2015.

Claudio Durán et al. Pioneering topological methods for network-based drug–target prediction by exploiting a brain-network self-organization theory. Briefings in Bioinformatics, 19(6):1183–1202, 2018.

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. Science, 286(5439):509–512, 1999.

Linyuan Lü, Ci-Hang Jin, and Tao Zhou. Similarity index based on local paths for link prediction of complex networks. Physical Review E, 80(4):046122, 2009.

Marinka Zitnik and Blaz Zupan. Data fusion by matrix factorization. IEEE Transactions on Pattern Analysis and Machine Intelligence, 37(1):41–53, 2015.

Bo Wang et al. Network enhancement as a general method to denoise weighted biological networks. Nature Communications, 9(1):1–8, 2018.

Linchuan Xu, Jiannong Cao, Xiaokai Wei, and Philip Yu. Network embedding via coupled kernelized multi-dimensional array factorization. IEEE TKDE, 2019.
Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In WWW, pages 1067–1077, 2015.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In NeurIPS, pages 1024–1034, 2017.

Tengfei Ma, Cao Xiao, Jiayu Zhou, and Fei Wang. Drug similarity integration through attentive multi-view graph auto-encoders. IJCAI, 2018.

Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. In NeurIPS, pages 5165–5175, 2018.

Masashi Tsubaki, Kentaro Tomii, and Jun Sese. Compound–protein interaction prediction with end-to-end learning of neural networks for graphs and sequences. Bioinformatics, 35(2):309–318, 2019.

Hakime Öztürk, Arzucan Özgür, and Elif Ozkirimli. Deepdta: deep drug–target binding affinity prediction. Bioinformatics, 34(17):i821–i829, 2018.

Yingkai Gao, Achille Fokoue, et al. Interpretable drug target prediction using deep neural representation. In IJCAI, pages 3371–3377, 2018.

Kexin Huang, Cao Xiao, Trong Nghia Hoang, Lucas M Glass, and Jimeng Sun. Caster: Predicting drug interactions with chemical substructure representation. AAAI, 2020.

Jae Yong Ryu, Hyun Uk Kim, and Sang Yup Lee. Deep learning improves prediction of drug–drug and drug–food interactions. PNAS, 115(18):E4304–E4311, 2018.

Feixiong Cheng and Zhongming Zhao. Machine learning-based prediction of drug–drug interactions by integrating drug phenotypic, therapeutic, chemical, and genomic properties. JAMIA, 21(2):e278–e286, 2014.

Tijana Milenković and Natasa Pržulj. Uncovering biological network function via graphlet degree signatures. Cancer Informatics, 6:1–8, 2007.

Wen Zhang, Xiang Yue, Weiran Lin, Wenjian Wu, Ruoqi Liu, Feng Huang, and Feng Liu. Predicting drug–disease associations by using similarity constrained matrix factorization. BMC Bioinformatics, 19(1):1–12, 2018.

Reza Ferdousi, Reza Safdari, and Yadollah Omidi. Computational prediction of drug-drug interactions based on drugs functional similarities. JBL, 70:54–64, 2017.

Ping Zhang, Fei Wang, Jianying Hu, and Robert Sorrentino. Label propagation prediction of drug-drug interactions based on clinical side effects. Scientific Reports, 5(1):1–10, 2015.

Marinka Zitnik and Jure Leskovec. Predicting multicellular function through multi-layer tissue networks. Bioinformatics, 33(14):i190–i198, 2017.

Yunan Luo, Xinbin Zhao, Jingtian Zhou, Jinglin Yang, Yanqing Zhang, Wenhua Kuang, Jian Peng, Ligong Chen, and Jianyang Zeng. A network integration approach for drug-target interaction prediction and computational drug repositioning from heterogeneous information. Nature Communications, 8(1):1–13, 2017.

Wenming Cao, Zhiyue Yan, Zhiquan He, and Zhihai He. A comprehensive survey on geometric deep learning. IEEE Access, 8:35929–35949, 2020.
Marinka Zitnik, Rok Sosič, Sagar Maheshwari, and Jure Leskovec. BioSNAP Datasets: Stanford biomedical network dataset collection, August 2018b.

Katja Luck et al. A reference map of the human protein interactome. bioRxiv, 2019.

Janet Piñero et al. The DisGeNET knowledge platform for disease genomics: 2019 update. Nucleic Acids Research, 11 2019. ISSN 0305-1048. doi: 10.1093/nar/gkz1021. URL https://doi.org/10.1093/nar/gkz1021. gkz1021.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. ICLR, 2014.

Fan RK Chung and Fan Chung Graham. Spectral graph theory. Number 92. American Mathematical Soc., 1997.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. JMLR, 9(Nov), 2008.

Shiva Shanker Reddy Mukku, PT Sivakumar, and Mathew Varghese. Clozapine use in geriatric patients—challenges. Asian Journal of Psychiatry, 33:63–67, 2018.

Sarah M Robertson et al. Clinical drug-drug interaction assessment of ivacaftor as a potential inhibitor of cytochrome p450 and p-glycoprotein. The Journal of Clinical Pharmacology, 55(1):56–62, 2015.

Pharmaceuticals DuPont. Product information. coumadin (warfarin). DuPont Pharmaceuticals, Wilmington, DE., 2000. URL https://www.medsafe.govt.nz/Profs/Datasheet/c/Coumadintab.pdf.

David S. Snyder. Interaction between Cyclosporine and Warfarin. Annals of Internal Medicine, 108(2):311–311, 02 1988. ISSN 0003-4819. doi: 10.7326/0003-4819-108-2-311_1. URL https://doi.org/10.7326/0003-4819-108-2-311_1.

& Company Inc Merck. Product information. belsomra (suvorexant). Merck & Company Inc, Whitehouse Station, NJ, 2014. URL https://www.drugs.com/drug-interactions/phenelzine-with-suvorexant-1839-0-3554-0.html?professional=1.

Pharmaceuticals Ligand. Product information. targretin (bexarotene). Ligand Pharmaceuticals, San Diego, CA., 1999. URL https://www.drugs.com/drug-interactions/bexarotene-with-deferasirox-378-0-791-0.html?professional=1.

Pharmaceuticals Novartis. Product information. clozaril (clozapine). Novartis Pharmaceuticals, East Hanover, NJ., 1989. URL https://www.drugs.com/drug-interactions/coumadin-with-zaponex-2311-1529-709-3273.html?professional=1, https://www.southernhealth.nhs.uk/EasysiteWeb/getresource.axd?AssetID=77706&type=full&servicetype=Inline.
A  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:

- **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.

Table A 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.

B  Details about baseline methods

- **L3** [Kovács et al., 2019] 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.
- **DeepWalk** [Perozzi et al., 2014] 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** [Grover and Leskovec, 2016] 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** [Ribeiro et al., 2017] 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.
Table 6: *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.

| 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 |
| 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 |
| 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 |
| 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 |

- **Spectral Clustering** [Tang and Liu, 2011] 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** [Kipf and Welling, 2016] applies variational graph auto-encoder and learns node embeddings that best reconstruct the adjacency 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** [Kipf and Welling, 2017] 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. SkipGNN usually converges within 5 epochs. The dropout rate is set to be 0.1.

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

## C  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 C.
Table 7: 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.

| 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    | DB0852      | BCHE            |
|      | 7    | CRO02       | 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     | MIPO1     |
|      | 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     |