LEARNING REPRESENTATIONS OF PARTIAL SUBGRAPHS BY SUBGRAPH INFOMAX

Anonymous authors
Paper under double-blind review

ABSTRACT

Subgraphs are important substructures of graphs, but learning their representations has not been studied well. Particularly, when we have partially observed subgraphs, existing node- or subgraph-level message-passing is likely to produce suboptimal representations. In this paper, we propose Intra- and Inter-Subgraph InfoMax, a model that learns subgraph representations under incomplete observation. Our model employs subgraph summaries at two different levels while maximizing the mutual information between the subgraph summaries and the node representations. By doing so, we reconstruct the representation of the underlying subgraph and improve its expressiveness from different angles of the local-global structure. We conduct experiments on three real-world datasets under training and evaluation protocols designed for this problem. Experimental results show that our model outperforms baselines in all settings.

1 INTRODUCTION

The graph neural network (GNN) has become a major framework to represent graph-structured data (Bronstein et al., 2017; Battaglia et al., 2018). GNNs have shown success in downstream tasks on nodes, edges, and graphs (Hu et al., 2020; Dwivedi et al., 2020). In addition to graphs, subgraphs can express various real-world data: information propagation graph in a social network, functional groups in a chemical compound, or disease in a knowledge graph of symptoms.

The current formulation of subgraph representation learning by Alsentzer et al. (2020) assumes full observation of nodes and edges in a subgraph, and that assumption often does not hold in real world problems of interest. For example, in the problem of fake news detection given its propagation tree, using a fully propagated subgraph would not be useful; instead, we would want to classify fake news with an early propagated subgraph before the news spreads to a large number of users as illustrated in Figure 1. If the assumption of complete observation is not met, existing models may learn suboptimal representations because of inaccurate message-passing with missing nodes or edges. This drawback occurs in message-passing not only between nodes but also between subgraphs and among the connected components in a subgraph. In this paper, we relax this assumption and learn representations of partially observed subgraphs.

For this 'partial subgraph learning' task, we propose the *Intra- and Inter-Subgraph InfoMax* (SGI) model based on local-global mutual information (MI) maximization inspired by Deep InfoMax (Hjelm et al., 2019) and its variants on graphs (Veličković et al., 2019; Sun et al., 2020). Similar to these previous models that maximize MI between the global summary (e.g., graph) and local parts (e.g., nodes), SGI maximizes MI between a subgraph and node representations. Through a GAN-like divergence MI estimator (Nowozin et al., 2016), SGI learns to distinguish for a specific subgraph whether a node belongs to the same subgraph (positive) or not (negative).

However, when nodes or edges are missing, their feature and structural information are lost. This makes summarizing the true subgraph with partial observations not straightforward. Thus, we create two different levels of subgraph summary for Intra- and Inter-SGI. First, Intra-SGI uses the observed subgraph summary to maximize the MI with nodes in the subgraph. Second, Inter-SGI assembles nodes of high MI with the observed subgraph using results of Intra-SGI, and reconstructs the subgraph with high MI nodes. Using the reconstructed subgraph allows Inter-SGI to obtain a summary closed to the full subgraph under insufficient observation. For the fake news early detection task as an example, Intra-SGI maximizes MI between the propagation tree of early spreaders and

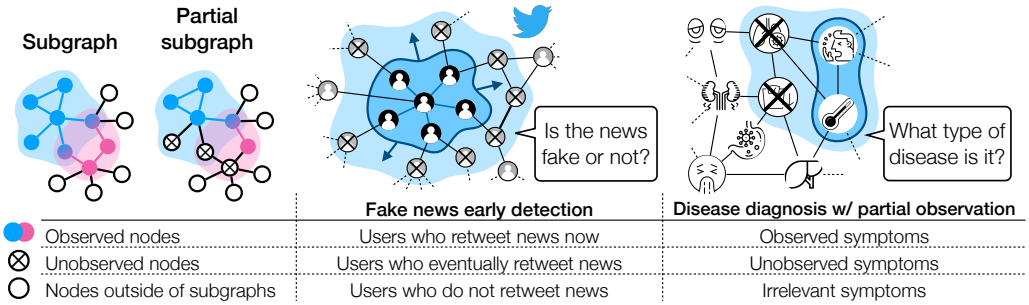


Figure 1: **Left:** Comparison of representation learning of subgraphs and partial subgraphs. The latter learns subgraph representations with partial observation of nodes and edges. **Middle & Right:** Real-world examples of partial subgraph learning: fake news early detection and disease diagnosis with partial observation. See **Real-world examples** in §2.2 for a more detailed description.

all spreaders of news. This can be interpreted as learning the probability of users propagating news, and with the probability, we can reconstruct subgraphs with users who are likely to propagate news. Inter-SGI makes the reconstructed subgraph to differentiate positive users who really spread the news and negative users who did not.

We demonstrate the improved representation learning performance of Intra- and Inter-SGI with experiments on three real-world datasets. These datasets emulate scenarios of fake news early detection, social network user profiling, and disease diagnosis with partial observation (Figure 1). Our model consistently outperforms baseline models for all datasets. We conduct additional analysis of the models' performance depending on the properties of the subgraphs and the global graph.

We present the following contributions. First, we formulate and characterize the problem of representation learning of partially observed subgraphs (§2). Second, we propose Intra- and Inter-SGI models that maximize MI between nodes and subgraphs for this problem (§3). Third, we present proper training and evaluation settings (§4) and demonstrate that our model outperforms baselines on three real-world datasets (§5). We make our code available for future research (`url_redacted`).

2 Partial Subgraph Learning Problem

We formulate the 'partial subgraph learning problem' and characterize it in terms of two properties: ordering in observation and relationship between edges in a subgraph and the global graph.

2.1 Problem Formulation

We first introduce notations used in the problem description. Let $\mathcal{G} = (\mathbb{V}^{\text{glob}}, \mathbb{A}^{\text{glob}}, \boldsymbol{X}^{\text{glob}})$ be a global graph, where $\mathbb{V}^{\text{glob}}$ is a set of nodes, $\mathbb{A}^{\text{glob}}$ is a set of edges, and $\boldsymbol{X}^{\text{glob}} \in \mathbb{R}^{|\mathbb{V}^{\text{glob}}| \times F^{\text{in}}}$ is an input feature matrix of nodes. A subgraph $\mathcal{S} = (\mathbb{V}^{\text{sub}}, \mathbb{A}^{\text{sub}})$ of $\mathcal{G}$ is defined as a graph, the nodes and edges of which are subsets of $\mathbb{V}^{\text{glob}}$ and $\mathbb{A}^{\text{glob}}$ respectively (i.e., $\mathbb{V}^{\text{sub}} \subset \mathbb{V}^{\text{glob}}$ and $\mathbb{A}^{\text{sub}} \subset \mathbb{A}^{\text{glob}}$). For a set of M subgraphs $\mathbb{S} = \{\mathcal{S}_1, ..., \mathcal{S}_M\}$ and corresponding labels $\{y_1, ..., y_M\}$ from 1 to C, Alsentzer et al. (2020) defined a subgraph prediction problem as learning representation $\boldsymbol{s} \in \mathbb{R}^F$ for each subgraph and its classifier $f : \mathbb{R}^F \to \{1, ..., C\}$, where F is the dimension of the representation. In addition, we summarize our notation in Table 3 in Appendix A.

We formulate the 'partial subgraph prediction problem' by relaxing the assumption of Alsentzer et al. (2020) on the complete observation. Specifically, we observe a subset of nodes or edges of the subgraph, as in Figure 1. Here, we define a partially observed subgraph (or the *partial subgraph*) $\mathcal{S}^{\text{obs}} = (\mathbb{V}^{\text{obs}}, \mathbb{A}^{\text{obs}})$ of the full subgraph $\mathcal{S}$ as a subgraph where $\mathbb{V}^{\text{obs}} \subset \mathbb{V}^{\text{sub}}$ and $\mathbb{A}^{\text{obs}} \subset \mathbb{A}^{\text{sub}}$. Like the subgraph prediction, given a set of partially observed subgraphs $\mathbb{S}^{\text{obs}} = \{\mathcal{S}_1^{\text{obs}}, ..., \mathcal{S}_M^{\text{obs}}\}$ and labels $\{y_1, ..., y_M\}$, our goal is to learn a representation $\boldsymbol{s} \in \mathbb{R}^F$ for $\mathcal{S}^{\text{obs}} \in \mathbb{S}^{\text{obs}}$ and its classifier $f : \mathbb{R}^F \to \{1, ..., C\}$. The degree of partial observation depends on the use case, but in this study, we assume only a few nodes are observed.

2.2 Properties

We discuss two important properties of this problem, which should be mainly considered in designing the evaluation protocol (§4.2) and models (§3, §4.2). Through this discussion, we provide guidelines on how to solve and evaluate our proposed task fairly.

Ordering in observation: ordered vs. unordered Our evaluation protocol focuses on selecting nodes in a partial subgraph ($\mathbb{V}^{\text{obs}}$) from the full subgraph ($\mathbb{V}^{\text{sub}}$). In §2.1, the only constraint is $\mathbb{V}^{\text{obs}} \subset \mathbb{V}^{\text{sub}}$. However, considering the dynamics and temporality in real-world graphs, there may be a specific ordering of observation of the nodes. Formally, when there exists node observation ordering as a bijective map $\pi : \mathbb{V}^{\text{sub}} \to \{1, ..., |\mathbb{V}^{\text{sub}}|\}$, we should choose $\mathbb{V}^{\text{obs}} = \{v | v \in \mathbb{V}^{\text{sub}}, \pi(v) \leq N^{\text{obs}}\}$ for fixed N^{obs} in evaluation sets, rather than choosing an arbitrary subset of $\mathbb{V}^{\text{sub}}$.

Relationship between edges in a subgraph and the global graph: identical vs. inclusive Since GNNs perform message-passing through edges, how we choose edges among two types, $\mathbb{A}^{\text{sub}}$ in subgraphs and $\mathbb{A}^{\text{glob}}$ in the global graph, is an important design decision. We know $\mathbb{A}^{\text{sub}} \subset \mathbb{A}^{\text{glob}}$ by definition, and there can be a special case in which the set of edges $\mathbb{A}^{\text{sub}}$ of the subgraph and the set of edges $\mathbb{A}^{\text{glob}}[\mathbb{V}^{\text{sub}}]$ of the induced subgraph are identical. We define the induced subgraph $\mathcal{G}[\mathbb{V}^{\text{sub}}]$ as $(\mathbb{V}^{\text{sub}}, \mathbb{A}^{\text{glob}}[\mathbb{V}^{\text{sub}}])$ where $\mathbb{A}^{\text{glob}}[\mathbb{V}^{\text{sub}}] = \{(v_i, v_j) | v_i, v_j \in \mathbb{V}^{\text{sub}}, (v_i, v_j) \in \mathbb{A}^{\text{glob}}\}$. In this case, the dataset will be referred to as identical ($\mathbb{A}^{\text{sub}} = \mathbb{A}^{\text{glob}}[\mathbb{V}^{\text{sub}}]$), otherwise inclusive ($\mathbb{A}^{\text{sub}} \subsetneq \mathbb{A}^{\text{glob}}[\mathbb{V}^{\text{sub}}]$). If we are dealing with the 'inclusive' case, it is necessary to decide which set of edges will be used between $\mathbb{A}^{\text{sub}}$ and $\mathbb{A}^{\text{glob}}[\mathbb{V}^{\text{sub}}]$.

Real-world examples We explain these properties in real-world scenarios. In a social network, information propagation graphs are *ordered* and *inclusive*. Since nodes are observed according to the order in which they are propagated, partial subgraphs should contain only early propagated nodes. Also, even if A follows B, A does not propagate all the information that B brings, so edges in the propagation graph and the social network may not be the same. Another example is the knowledge graph of symptoms and subgraphs which represent the diseases, and the subgraphs in this example are *unordered* and *identical*. Missing symptoms because of problems such as equipment errors would not occur in any specific order. And prior medical knowledge determines the edges connecting symptoms in the global graph, and the subgraph would inherit the same set of edges.

3 Intra- and Inter-Subgraph InfoMax

This section describes Intra- and Inter-Subgraph InfoMax (SGI) design and how they are combined in an end-to-end pipeline.

3.1 Notations and Background: Graph Neural Encoder and Readout

Most models that perform graph-level prediction consist of the encoder $\mathcal{E}$ and readout function $\mathcal{R}$. The node representations $\boldsymbol{H} = \{\boldsymbol{h}_1, ..., \boldsymbol{h}_N\} = \mathcal{E}^{\text{in}}(\boldsymbol{X}, \mathbb{A}) \in \mathbb{R}^{N \times F}$ are computed with an encoder $\mathcal{E}^{\text{in}} : \mathbb{R}^{N \times F^{\text{in}}} \times \mathbb{A} \to \mathbb{R}^{N \times F}$ that consists of multiple graph neural layers. Node representations are summarized as a fixed-size vector $\boldsymbol{s} = \mathcal{R}(\boldsymbol{H}) \in \mathbb{R}^F$ with a readout function $\mathcal{R} : \mathbb{R}^{N \times F} \to \mathbb{R}^F$. Many studies have proposed various readout (or graph pooling) functions, including relatively simple methods that apply sum, max, or mean operations on node features (Cangea et al., 2018).

As in §2.1, we use the superscript to denote the graph type to which matrices and vectors belong; for example, $\mathbb{V}^{\text{obs}}$ denotes the observed nodes, $\boldsymbol{H}^{\text{obs}}$ and $\boldsymbol{s}^{\text{obs}}$ denote the feature matrix and the summary vector for the observed nodes, respectively.

3.2 Local-Global Mutual Information Maximization for Partial Subgraphs

The partial subgraph learning problem has a structural hierarchy from the global graph to full and partial subgraphs. We employ a local-global mutual information (MI) maximization method to learn this structural information without additional labeling costs.

In detail, our model maximizes MI between the summary vector $\boldsymbol{s}$ of the subgraph (i.e., global representation) and the node representation $\boldsymbol{h}$ in the subgraph (i.e., local regions of the input) (Hjelm et al., 2019). Among several MI estimators modeled with neural networks (Belghazi et al., 2018; Oord et al., 2018), we choose the GAN-like divergence MI estimator (Nowozin et al., 2016). To maximize this estimator, we maximize the following loss between samples from the joint distribution $P_{s,v}$ and the product of marginal distributions $P_s \times P_{\tilde{v}}$, that is,

$$\mathcal{L} = \mathbb{E}_{P_{s,v}} \left[\log \mathcal{D}\left(\boldsymbol{h}_v, \boldsymbol{s}_i\right)\right] + \mathbb{E}_{P_s \times P_{\tilde{v}}} \left[\log\left(1 - \mathcal{D}(\boldsymbol{h}_{\tilde{v}}, \boldsymbol{s}_i)\right)\right], \tag{1}$$

where $\boldsymbol{h}_v$ and $\boldsymbol{s}_i$ are representations corresponding to node v and the subgraph i respectively, and $\mathcal{D} : \mathbb{R}^F \times \mathbb{R}^F \to [0, 1]$ is a discriminator that computes the score reflecting how much $\boldsymbol{h}_v$ and $\boldsymbol{s}$ are related to each other. Note that the discriminator $\mathcal{D}$ is activated by point-wise sigmoid σ.

With only a few observed nodes, a naive summary based on those few does not hold much information of the entire subgraph. This is why existing InfoMax models on graphs and nodes are suboptimal for this problem. Instead, we design a pipeline that reconstructs a full subgraph based on observed nodes with two-level InfoMax losses. It allows the final subgraph summary to encode the observed nodes and important neighbors around them. In the following two paragraphs, we describe the details and rationale of how we summarize subgraph representation and which negative samples are used for each SGI.

Intra-Subgraph InfoMax Intra-SGI maximizes the MI between the node representations in the subgraph and the summary vector $\boldsymbol{s}^{\text{obs}}$ of the observed nodes. The summary $\boldsymbol{s}^{\text{obs}}$ is created by applying an additional encoder $\mathcal{E}^Q$ to observed node representations $\boldsymbol{H}^{\text{obs}}$ and then readout $\mathcal{R}^{\text{obs}}$.

$$\boldsymbol{H}^{Q,\text{obs}} = \mathcal{E}^Q(\boldsymbol{H}^{\text{obs}}),\ \boldsymbol{s}^{\text{obs}} = \mathcal{R}^{\text{obs}}(\boldsymbol{H}^{Q,\text{obs}}). \tag{2}$$

We want to maximize the MI between $\boldsymbol{s}^{\text{obs}}$ and nodes in the subgraph. However, we cannot know the exact boundary of the subgraph at this stage; the model only knows which nodes are included in the observed subgraph but not the full subgraph. Using all nodes in the global graph $\mathcal{G}$ can solve this issue, but it is generally huge that we can only load a part of it into the GPU memory. There can be various heuristics to select nodes among $\mathbb{V}^{\text{glob}}$ that are likely to belong to the subgraph $\mathbb{V}^{\text{sub}}$, and we take a simple way to sample the k-hop neighbors $\mathcal{N}^k(u)$ of observed nodes $u \in \mathbb{V}^{\text{obs}}$. There are two kinds of nodes in k-hop neighbors: nodes that are actually included in the subgraph $\mathbb{V}^{\text{sub}_k} = \bigcup_{u\in\mathbb{V}^{\text{obs}}} \mathcal{N}^k(u) \cap \mathbb{V}^{\text{sub}}$ and nodes that are not, $\mathbb{V}^{\text{glob}_k} = \bigcup_{u\in\mathbb{V}^{\text{obs}}} \mathcal{N}^k(u) \cap (\mathbb{V}^{\text{glob}} \setminus \mathbb{V}^{\text{sub}})$. We use these two sets separately to describe the forward pass of training, but the model in the inference stage cannot distinguish them. These nodes in $\mathbb{V}^{\text{obs}}$, $\mathbb{V}^{\text{sub}_k}$ and $\mathbb{V}^{\text{glob}_k}$ are encoded to $\boldsymbol{H}^{K,*}$ with $\mathcal{E}^{\text{in}}$ and $\mathcal{E}^K$, that is

$$\boldsymbol{H}^{K,*} = \{\boldsymbol{h}_1^K, ..., \boldsymbol{h}_{|\mathbb{V}^{\text{obs}}|+|\mathbb{V}^{\text{sub}_k}|+|\mathbb{V}^{\text{glob}_k}|}^K\} = \mathcal{E}^K(\boldsymbol{H}^*) \text{ where } \boldsymbol{H}^* = \left[\boldsymbol{H}^{\text{obs}} \parallel \boldsymbol{H}^{\text{sub}_k} \parallel \boldsymbol{H}^{\text{glob}_k}\right]. \tag{3}$$

Following Equation 1, we propose the Intra-SGI loss that maximizes the MI between $\boldsymbol{H}^{K,*}$ and $\boldsymbol{s}^{\text{obs}}$ by using nodes in $\mathbb{V}^{\text{obs}} \cup \mathbb{V}^{\text{sub}_k}$ as positive samples and nodes in $\mathbb{V}^{\text{glob}_k}$ as negative samples. Note that $\mathbb{V}^{\text{sub}_k}$ and $\mathbb{V}^{\text{glob}_k}$ are subsets of $\mathbb{V}^{\text{sub}}$ and $\mathbb{V}^{\text{glob}}$, respectively, but if k is smaller than the diameter of $\mathcal{S}$ and $\mathcal{G}$, they cannot be guaranteed to be the same. The score $\mathcal{D}^{\text{obs}}(\boldsymbol{h}_v, \boldsymbol{s}^{\text{obs}})$ can be interpreted as modeling the probability that the node belongs to the subgraph. The Intra-SGI's objective to maximize this probability implies reconstructing the true structure from the corrupted subgraph. In that sense, this reconstruction resembles the denoising auto-encoder (Vincent et al., 2008) and the graph auto-encoder (Kipf & Welling, 2016).

$$\mathcal{L}^{\text{intra}} = \frac{1}{|\mathbb{V}^{\text{obs}}\cup\mathbb{V}^{\text{sub}_k}|+|\mathbb{V}^{\text{glob}_k}|} \left[\sum_{v\in\mathbb{V}^{\text{obs}}\cup\mathbb{V}^{\text{sub}_k}} \log \mathcal{D}^{\text{obs}}\left(\boldsymbol{h}_v^K, \boldsymbol{s}^{\text{obs}}\right) + \sum_{\tilde{v}\in\mathbb{V}^{\text{glob}_k}} \log\left(1 - \mathcal{D}^{\text{obs}}(\boldsymbol{h}_{\tilde{v}}^K, \boldsymbol{s}^{\text{obs}})\right)\right]. \tag{4}$$

Here, the negative nodes $\mathbb{V}^{\text{glob}_k}$ are not sampled from the true marginal distribution. Intuitively, this $\mathbb{V}^{\text{glob}_k}$, a set of nodes closely linked to the subgraph within k-hop, can be considered hard negative samples conditioned on positive samples. This approach is known to learn a better representation in contrastive and metric learning (Schroff et al., 2015; Oh Song et al., 2016; Oord et al., 2018; Zhuang et al., 2019), but such non-i.i.d sampling may break the assumption on the MI bound (Tschannen et al., 2020). In Proposition 1, similar to Conditional-NCE (CNCE) (Wu et al., 2021), we prove that a specific choice of negative sample distribution forms the lower bound of the GAN-like divergence MI estimator.

Proposition 1 (The conditional GAN-like divergence MI bound)**.** *For d-dimensional random variables X and Y with a joint distribution $p(x, y)$ and marginal distributions $p(x)$ and $p(y)$, fix any function $f : (X, Y) \to \mathbb{R}$ and realization x of X. Let $c_x = \mathbb{E}_{y\sim p(y)}\left[e^{f(x,y)}\right]$, $\mathbb{B}_{c_x} \subset \mathbb{R}$ be strictly lower bounded by c_x, and $\mathbb{Y}_{c_x} = \{y | e^{f(x,y)} \in \mathbb{B}_{c_x}\}$ with an assumption of $p(\mathbb{Y}_{c_x}) > 0$. For $\mathbb{Y}_r$ in the Borel σ-algebra over $\mathbb{R}^d$, let $q(Y \in \mathbb{Y}_r | X = x) = p(\mathbb{Y}_r | \mathbb{Y}_{c_x})$, then $\mathcal{I}^{CGAN} \leq \mathcal{I}^{GAN}$ where*

$$\mathcal{I}^{CGAN} = \mathbb{E}_{x,y\sim p(x,y)}\left[\log\sigma(f(x,y))\right] + \mathbb{E}_{x\sim p(x)}\mathbb{E}_{y\sim q(y|x)}\left[\log\left(1-\sigma(f(x,y))\right)\right], \tag{5}$$

$$\mathcal{I}^{GAN} = \mathbb{E}_{x,y\sim p(x,y)}\left[\log\sigma(f(x,y))\right] + \mathbb{E}_{x\sim p(x)}\mathbb{E}_{y\sim p(y)}\left[\log\left(1-\sigma(f(x,y))\right)\right]. \tag{6}$$

The proof is given in Appendix B. After applying f to the training set, the CNCE uses a subset, the exponentiated similarity $e^{f(\cdot,\cdot)}$ of which is bigger than that of a certain percentile. Instead, we employ k-hop sampling, which uses hop distance as a proxy of embedding distance (or dissimilarity). This method assumes that the hop and embedding distances of nodes created by message-passing are highly correlated. It is more efficient than using the actual similarity since it does not have to evaluate f for all instances.

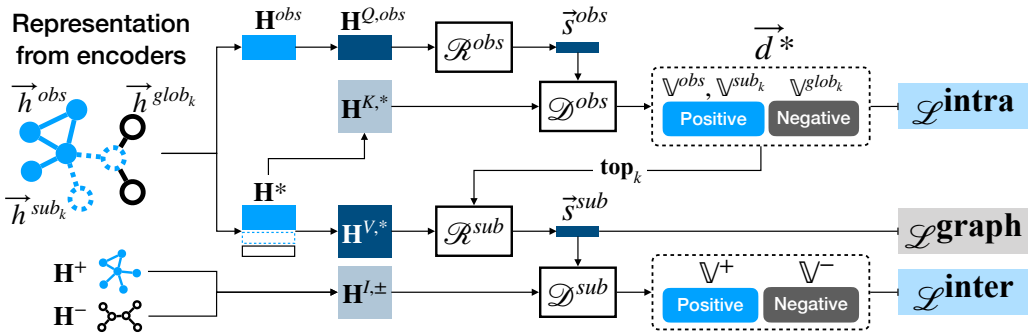


Figure 2: Overview of Intra- and Inter-SGI. Given node representations $\boldsymbol{h}$, both SGI methods maximize MI between the summarized representation of the subgraph and its nodes. (1) Intra-SGI uses the observed subgraph representation $\boldsymbol{s}^{\text{obs}}$ as the summary vector and k-hop neighbors that are not part of the subgraph $\mathbb{V}^{\text{glob}_k}$ as negative samples. (2) Inter-SGI uses the reconstructed subgraph representation $\boldsymbol{s}^{\text{sub}}$ as the summary vector and nodes from other subgraphs $\mathbb{V}^-$ as negative samples.

Inter-Subgraph InfoMax Inter-SGI maximizes the MI between the node representations in the subgraph and the summary vector $\boldsymbol{s}^{\text{sub}}$ of the reconstructed subgraph using the score from *Intra*-SGI. Specifically, given the score vector $\boldsymbol{d}^* \in \mathbb{R}^{|\mathbb{V}^{\text{obs}}|+|\mathbb{V}^{\text{sub}_k}|+|\mathbb{V}^{\text{glob}_k}|}$ computed by the discriminator $\mathcal{D}^{\text{obs}}$ on $\boldsymbol{H}^{K,*}$ and $\boldsymbol{s}^{\text{obs}}$, we create $\boldsymbol{s}^{\text{sub}}$ by pooling nodes corresponding to the top-k value of $\boldsymbol{d}^*$. Formally,

$$\boldsymbol{d}^* = \mathcal{D}^{\text{obs}}\left(\boldsymbol{H}^{K,*}, \boldsymbol{s}^{\text{obs}}\right), \ \texttt{idx} = \text{top}_k(\boldsymbol{d}^*), \ \boldsymbol{H}^{V,*} = \mathcal{E}^V(\boldsymbol{H}^*), \ \boldsymbol{s}^{\text{sub}} = \mathcal{R}^{\text{sub}}(\boldsymbol{H}^{V,*}_{[\texttt{idx}]}, \boldsymbol{d}^*_{[\texttt{idx}]}). \quad (7)$$

For positive samples, Inter-SGI uses all nodes in the subgraph $\mathcal{S}^+ := \mathcal{S}$. Thus, interactions between the summary $\boldsymbol{s}^{\text{sub}}$ and all nodes in the subgraph $\mathbb{V}^+ := \mathbb{V}^{\text{sub}}$ can be modeled. This cannot be done in Intra-SGI since its positive nodes $\mathbb{V}^{\text{sub}_k}$ is equal to $\mathbb{V}^{\text{sub}}$ only for diameter k, but we can only choose a small enough k to fit in the GPU memory. We use nodes in another subgraph $\mathcal{S}^- = (\mathbb{V}^-, \mathbb{A}^-)$ drawn from the dataset $\mathbb{S}$ for negative samples. We first encode nodes in $\mathbb{V}^+, \mathbb{V}^-$ to $\boldsymbol{H}^+, \boldsymbol{H}^-$ with $\mathcal{E}^{\text{in}}$, and here, input edges can be either $\mathbb{A}^{\text{sub}}$ or $\mathbb{A}^{\text{glob}}[\mathbb{V}^{\text{sub}}]$ (§2.2). Then, they are encoded by $\mathcal{E}^I$ and plugged with $\boldsymbol{s}^{\text{sub}}$ and another discriminator $\mathcal{D}^{\text{sub}}$ into Equation 1 to form the Inter-SGI loss.

$$\boldsymbol{H}^{I,\pm} = \{\boldsymbol{h}^I_1, ..., \boldsymbol{h}^I_{|\mathbb{V}^+|+|\mathbb{V}^-|}\} = \mathcal{E}^I(\boldsymbol{H}^\pm) \quad \text{where} \quad \boldsymbol{H}^\pm = \left[\boldsymbol{H}^+ \,\|\, \boldsymbol{H}^-\right]. \quad (8)$$

$$\mathcal{L}^{\text{inter}} = \frac{1}{|\mathbb{V}^+|+|\mathbb{V}^-|}\left[\sum_{v\in\mathbb{V}^+} \log \mathcal{D}^{\text{sub}}\left(\boldsymbol{h}^I_v, \boldsymbol{s}^{\text{sub}}\right) + \sum_{\tilde{v}\in\mathbb{V}^-} \log\left(1 - \mathcal{D}^{\text{sub}}(\boldsymbol{h}^I_{\tilde{v}}, \boldsymbol{s}^{\text{sub}})\right)\right]. \quad (9)$$

3.3 Overview of Intra- and Inter-SGI

We summarize the entire pipeline of the Intra- and Inter-SGI model in Figure 2 and explain it below. Given a global graph $\mathcal{G} = (\mathbb{V}^{\text{glob}}, \mathbb{A}^{\text{glob}}, \boldsymbol{X}^{\text{glob}})$ and a partially observed subgraph $\mathcal{S}^{\text{obs}} = (\mathbb{V}^{\text{obs}}, \mathbb{A}^{\text{obs}})$ of the full (positive) subgraph $\mathcal{S} := \mathcal{S}^+ = (\mathbb{V}^{\text{sub}}, \mathbb{A}^{\text{sub}})$ from the dataset $\mathbb{S}$,

1. **Sampling:** Sample a k-hop subgraph $(\mathbb{V}^*, \mathbb{A}^{\text{glob}}[\mathbb{V}^*])$ of $\mathbb{V}^{\text{obs}}$ where $\mathbb{V}^* = \mathbb{V}^{\text{obs}} \cup \mathbb{V}^{\text{sub}_k} \cup \mathbb{V}^{\text{glob}_k}$ are k-hop neighbors of $\mathbb{V}^{\text{obs}}$. And draw a negative subgraph $\mathcal{S}^- = (\mathbb{V}^-, \mathbb{A}^-)$ from $\mathbb{S}$.
2. **Encoding:** Encode $\boldsymbol{H}^* = \mathcal{E}^{\text{in}}(\boldsymbol{X}^*, \mathbb{A}^{\text{glob}}[\mathbb{V}^*])$ and $\boldsymbol{H}^\pm = \mathcal{E}^{\text{in}}(\boldsymbol{X}^\pm, \mathbb{A}^{\text{sub},\pm}$ or $\mathbb{A}^{\text{glob}}[\mathbb{V}^{\text{sub},\pm}])$ using a graph neural encoder $\mathcal{E}^{\text{in}}$ where $\boldsymbol{X}^*$ and $\boldsymbol{X}^\pm$ are rows of $\boldsymbol{X}$ corresponding to $\mathbb{V}^*$ and $\mathbb{V}^\pm$. Then, get $\boldsymbol{H}^{Q,\text{obs}}, \boldsymbol{H}^{K,*}, \boldsymbol{H}^{V,*}, \boldsymbol{H}^{I,\pm}$ with $\mathcal{E}^Q, \mathcal{E}^K, \mathcal{E}^V, \mathcal{E}^I$ on $\boldsymbol{H}^{\text{obs}}, \boldsymbol{H}^*, \boldsymbol{H}^*, \boldsymbol{H}^\pm$.
3. **Intra-SGI loss ($\mathcal{L}^{\text{intra}}$):** Summarize $\boldsymbol{H}^{Q,\text{obs}}$ to the observed subgraph representation $\boldsymbol{s}^{\text{obs}}$ and get $\mathcal{L}^{\text{intra}}$ with node representations $\left[\boldsymbol{H}^{K,\text{obs}} \,\|\, \boldsymbol{H}^{K,\text{sub}_k}\right]$ (positive) and $\boldsymbol{H}^{K,\text{glob}_k}$ (negative).
4. **Inter-SGI loss ($\mathcal{L}^{\text{inter}}$):** Summarize $\boldsymbol{H}^{V,*}$ to the reconstructed subgraph representation $\boldsymbol{s}^{\text{sub}}$ with the score $\boldsymbol{d}^*$ from Inter-SGI loss. Compute $\mathcal{L}^{\text{inter}}$ with node representations $\boldsymbol{H}^{I,+}$ (positive) and $\boldsymbol{H}^{I,-}$ (negative).
5. **Loss on graph labels ($\mathcal{L}^{\text{graph}}$):** Compute the logit vector $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{s}^{\text{sub}}$ with a feed-forward neural network parametrized by $\boldsymbol{W} \in \mathbb{R}^{C\times F}$. Then, compute the cross-entropy loss $\mathcal{L}^{\text{graph}}$ on the graph label y. If the subgraph has a subgraph-level feature $\boldsymbol{g}$, we first transform it to the vector of the same length of $\boldsymbol{s}^{\text{sub}}$, then feed them to the last layer together, that is $\boldsymbol{y} = \boldsymbol{W}\left[\boldsymbol{s}^{\text{sub}} \,\|\, \mathcal{E}^G(\boldsymbol{g})\right]$ where $\boldsymbol{W} \in \mathbb{R}^{C\times 2F}$. Bias terms are omitted for simplicity.
6. Combine all losses, $\mathcal{L}^{\text{graph}}$, $\mathcal{L}^{\text{intra}}$, and $\mathcal{L}^{\text{inter}}$, and update the parameters (including $\boldsymbol{W}$) of all $\mathcal{E}$, $\mathcal{R}$, and $\mathcal{D}$ to maximize $-\mathcal{L}^{\text{graph}} + \lambda^{\text{intra}}\mathcal{L}^{\text{intra}} + \lambda^{\text{inter}}\mathcal{L}^{\text{inter}}$.

4 EXPERIMENTS

We describe the experimental settings: three real-world datasets, training and evaluation protocols, and detailed model architectures including the baseline models for comparison.

4.1 DATASETS

We experiment with three real-world datasets: FNTN (ordered, inclusive) (Kim et al., 2019), EM-User (unordered, identical) (Alsentzer et al., 2020), and HPO-Metab (unordered, identical) (Alsentzer et al., 2020). FNTN consists of a follower network on Twitter ($\mathcal{G}$), propagation trees of news ($\mathbb{S}$), and contents ($\boldsymbol{g}$). EM-User consists of a fitness network of workouts ($\mathcal{G}$) and subgraphs ($\mathbb{S}$) corresponding to users. The global graph $\mathcal{G}$ of HPO-Metab is a knowledge graph of symptoms of rare diseases. Each subgraph $\mathcal{S}$ is a collection of symptoms associated with a metabolic disorder. In Appendix C, we describe the tasks corresponding to the datasets, detailed data statistics (Table 4), pre-processing steps, and splits.

4.2 EXPERIMENTAL SET-UP

Partial subgraph generation We generate partial subgraphs by sampling nodes from the full subgraphs. For the ordered FNTN dataset, we use the early nodes in the propagation subgraph. For other unordered datasets, we uniformly sample a fixed number of nodes from the subgraph.

Training and evaluation settings We fix validation and test node sets with the *constant* size for each subgraph. This is a more realistic setting than selecting nodes proportional to the subgraph size (i.e., $|\mathbb{V}^{\text{sub}}|$) in that we cannot know the exact size at the evaluation stage. For the training set, we may or may not include all nodes. If we include them, the entire structure of the subgraph can be used for training, but there could be a distribution shift between training and test sets.

Based on the above rationale, we carry out this study with the following three training/evaluation settings.

1. **constant/constant**: For both training and evaluation sets, we set the number of observed nodes $|\mathbb{V}^{\text{obs}}|$ to 4 for HPO-Metab, where the average number of nodes is fewer than 16, and 8 for FNTN and EM-User, both with average size of subgraphs much greater than 16. Further, to see how the performance changes with $|\mathbb{V}^{\text{obs}}|$, we conduct experiments where $|\mathbb{V}^{\text{obs}}|$ is 8, 16, 32, and 64 for datasets with an average number of nodes greater than 64: FNTN and EM-User.
2. **100%/constant**: In this setting, we train with complete subgraphs, and evaluate with subgraphs cut into a constant size (4 or 8). As the SGI model is designed to learn representations with partial subgraphs, we experiment with this setting only with the baseline models.
3. **100%/100%**: We experiment with training and evaluation sets where we can observe all of the nodes and edges (i.e., 100%). We can think of this setting as the oracle for these tasks.

Model and training details For the graph neural encoder $\mathcal{E}^{\text{in}}$ in all experiments, we use the two-layer model with skip connections (He et al., 2016). We stack GraphSAGE (Hamilton et al., 2017) layers known to work well in the inductive setting. As an input of $\mathcal{E}^{\text{in}}$, the features of nodes $\boldsymbol{X}^{\text{glob}} \in \mathbb{R}^{|\mathbb{V}^{\text{glob}}| \times F^{\text{in}}}$ are trainable parameters with F^{in} of 32 (FNTN) and 64 (others). For HPO-Metab and EM-User, we use pre-trained embeddings from Alsentzer et al. (2020). For directed graphs in FNTN, we use a bi-directional encoder similar to Bi-RNNs (Schuster & Paliwal, 1997).

We use the two-layer Transformer (Vaswani et al., 2017) and the soft-attention pooling (Li et al., 2015) parametrized by $\boldsymbol{w} \in \mathbb{R}^{1 \times F}$ for $\mathcal{E}^{Q}$ and $\mathcal{R}^{\text{obs}}$ in Equation 2. We add the positional encoding (PE) before Transformer if the task has an ordering in the observation (§2.2).

$$\mathcal{E}^{Q}(\boldsymbol{H}) = \text{Transformer}(\boldsymbol{H}[+\text{PE}]),\ \mathcal{R}^{\text{obs}}(\boldsymbol{H}) = \text{softmax}(\boldsymbol{w}(\boldsymbol{H})^{\top})\boldsymbol{H}. \tag{10}$$

The other encoders denoted with $\mathcal{E}$ are MLPs, and $\mathcal{E}^{G}$ is a single-layer, and the rest are two-layer models. In the case of $\mathcal{R}^{\text{sub}}$, we employ the weighted sum of row vectors in $\boldsymbol{H}$ by the score $\boldsymbol{d}$ like top-k pooling (Gao & Ji, 2019; Knyazev et al., 2019).

$$\mathcal{E}^{K \text{ or } V \text{ or } I \text{ or } G}(\boldsymbol{H}) = \text{MLP}(\boldsymbol{H}),\ \mathcal{R}^{\text{sub}}(\boldsymbol{H}, \boldsymbol{d}) = \text{softmax}(\boldsymbol{d})\boldsymbol{H}. \tag{11}$$

Table 1: Mean and standard deviation of the classification accuracy of 5 runs. The first two columns represent the number of observed nodes used in the evaluation and training of each experiment (See §4.2 for more detail). If the number of observed nodes is constant, the setting corresponding to each dataset is indicated with daggers (†, ‡). Results of the unpaired *t*-test with the best baseline (except for the oracle) are denoted by asterisks ($^{**}p$-value $< .001$, $^{*}p$-value $< .05$).

Observed Nodes		Model	Dataset		
Evaluation	Training		FNTN‡	EM-User‡	HPO-Metab†
100%	100%	GraphSAGE	$86.3_{\pm 0.7}$	$82.1_{\pm 1.2}$	$47.7_{\pm 3.3}$
$8^{\ddagger}, 4^{\dagger}$	100%	MLP	$83.7_{\pm 1.5}$	$71.5_{\pm 3.6}$	$36.0_{\pm 4.1}$
		GraphSAGE	$85.9_{\pm 1.0}$	$68.5_{\pm 3.2}$	$35.3_{\pm 2.5}$
		SubGNN	N/A	$72.3_{\pm 3.0}$	$39.0_{\pm 1.8}$
	$8^{\ddagger}, 4^{\dagger}$	MLP	$82.5_{\pm 2.6}$	$71.9_{\pm 4.6}$	$43.5_{\pm 4.4}$
		GraphSAGE	$84.9_{\pm 1.3}$	$68.1_{\pm 2.6}$	$44.1_{\pm 1.3}$
		SubGNN	N/A	$61.3_{\pm 5.4}$	$37.1_{\pm 1.5}$
		Intra-SGI	$87.7^{*}_{\pm 1.0}$	$77.4^{*}_{\pm 3.2}$	$43.8_{\pm 2.3}$
		Inter-SGI	$87.3^{*}_{\pm 0.0}$	$75.7_{\pm 3.9}$	$47.1^{*}_{\pm 2.1}$
		Intra/Inter-SGI	$89.6^{**}_{\pm 1.0}$	$77.0^{*}_{\pm 2.8}$	$44.6_{\pm 1.6}$

Our discriminators ($\mathcal{D}^{\text{obs}}$ and $\mathcal{D}^{\text{sub}}$) are bilinear scoring function parametrized by $\boldsymbol{W} \in \mathbb{R}^{F \times F}$ and $\boldsymbol{b} \in \mathbb{R}$, similar to other InfoMax models (Veličković et al., 2019; Oord et al., 2018).

$$\mathcal{D}^{\text{obs or sub}}(\boldsymbol{h}, \boldsymbol{s}) = \sigma(\boldsymbol{h}^{\top} \boldsymbol{W} \boldsymbol{s} + \boldsymbol{b}). \tag{12}$$

All models, including baselines, use $F = 64$ features, the ReLU activation, feature dropout of 0.2 (Srivastava et al., 2014), and are trained by the Adam optimizer (Kingma & Ba, 2014) with the learning rate of 10^{-3}. For k-hop sampling in SGIs, we sample nodes in a one-hop neighborhood for input (i.e., $k = 1$). Details of model, training, and hyperparameters are in Appendix D, and ablation study on architectures and hyperparameters is in Appendix E.

Model variants There are three ways to use our models: Intra-SGI, Inter-SGI, or both. Using Intra-SGI alone is equivalent to the entire pipeline without $\mathcal{D}^{\text{sub}}$. However, using only Inter-SGI requires to re-define $\boldsymbol{s}^{\text{sub}}$ without the outcome of Intra-SGI. In this case, we apply the mean pooling after the two-layer MLP (i.e., $\mathcal{R}^{\text{sub}}(\boldsymbol{H}) = \frac{1}{N}\sum_{i=1}^{N} \text{MLP}(\boldsymbol{h}_i)$) to $\boldsymbol{H}^{\text{obs}}$ to replace $\boldsymbol{s}^{\text{sub}}$.

Baselines All baselines share the following encoder-readout architecture:

$$\boldsymbol{H} = \mathcal{E}^{B}(\boldsymbol{X}, \mathbb{A}),\ \boldsymbol{s}^{B} = \mathcal{R}^{B}(\boldsymbol{H}),\ \boldsymbol{y} = \boldsymbol{W}\boldsymbol{s}^{B} \text{ or } \boldsymbol{W}\left[\boldsymbol{s}^{B} \parallel \mathcal{E}^{G}(\boldsymbol{g})\right], \tag{13}$$

where $\mathcal{E}^{B}$ is the residual two-layer model of MLP, GraphSAGE (Hamilton et al., 2017), and SubGNN (Alsentzer et al., 2020). We set $\mathcal{R}^{B}$ the two-layer MLP after mean pooling following the original implementation for SubGNN and mean pooling after two-layer MLP for others. All baselines have the same number of layers as our model.

SubGNN requires precomputing shortest path distances between all node pairs, and this has a high complexity of time $O(|\mathbb{V}^{\text{glob}}||\mathbb{A}^{\text{glob}}| + |\mathbb{V}^{\text{glob}}|^2)$ and space $O(|\mathbb{V}^{\text{glob}}|^2)$. It makes training on FNTN dataset ($\times 24.8$ more nodes for HPO-Metab) impractical in terms of computation and memory (1TB of the matrix); thus, we report the performance of SubGNN only on EM-User and HPO-Metab.

5 Results and Discussions

In this section, we report the experimental results of our three SGI variants and baselines. In addition to the performance comparison between models, we demonstrate the performance change with regard to the number of observed nodes and the choice of edge type.

Performance by models and datasets Table 1 summarizes the accuracies of various models on the datasets, each averaged over five runs. We confirm that SGI outperforms all comparison models for all three datasets with statistical significance, except for Intra-SGI on HPO-Metab.

Interestingly, how Intra- and Inter-SGI contribute to the performance improvement varies depending on the dataset. Both models improve accuracy on FNTN and when used together, Intra/Inter-SGI

even surpasses the oracle (100%/100%) with a margin of 3.3%p. We interpret this as our model learns useful signals in the k-hop neighborhood $\mathbb{V}^{\text{glob}_k}$ as well as those in the full subgraph $\mathbb{V}^{\text{sub}}$. Specifically, SGIs can leverage nodes in a Twitter network (i.e., users) who are interested in the news but have not explicitly retweeted it. In EM-User, both improve performance as well, but there is no significant difference between the three SGI models (p-value of 0.71 in one-way ANOVA). In HPO-Metab, Intra-SGI rather degrades performance; however, with only Inter-SGI, it shows almost the same performance as the oracle with a difference of 0.6%p. We argue that this is caused by differences in the characteristics of the global graph and subgraphs.

First, the density of the global graph affects the performance of discriminator $\mathcal{D}^{\text{obs}}$ in Intra-SGI. Note that the higher density, the more neighbor nodes are included in k-hop neighbor sampling. Since most of the sampled neighbors are not in the subgraph, it is difficult to distinguish which of the large number of neighbors belong to the subgraph based on $\mathcal{D}^{\text{obs}}$. This is why Intra-SGI underperforms on HPO-Metab, which has a higher density (0.03) than FNTN (2×10^{-4}) and EM-User (2.8×10^{-3}). On the contrary, Inter-SGI, without the neighborhood noise, shows high performance on HPO-Metab. The noise from high density is still relevant in using Intra- and Inter-SGI together, and makes it perform worse than Inter-SGI alone.

Second, we can see the benefit of both Intra- and Inter-SGI when k-hop and full subgraphs provide strongly different views (or a hard contrastive task) (Chen et al., 2020; You et al., 2020; Tian et al., 2020). In FNTN, there are different patterns between the initial and the final propagation trees of fake news (Murayama et al., 2021). Therefore, the effect of contrastive learning in Intra- and Inter-SGI offer sufficiently different views, resulting in higher performance than using them independently. On the other hand, for EM-User with uniform partial observations, k-hop and full subgraphs provide similar views, and there is no significant performance difference between SGI variants. See Appendix F for detailed analysis.

Performance of baselines by settings Since we present a new class of problems, we discuss the behaviors of representative baselines. We make three observations with Table 1. First, SubGNN shows better results in 100%/constant than the constant/constant setting. SubGNN generalizes well when the model can use all information in subgraphs, as message-passing is performed between subgraphs. Second, for HPO-Metab, all models except SubGNN perform worse when training with all nodes. Even in SubGNN, the improvement of 2.9%p is relatively small (p-value of 0.15 in t-test) considering that this model shows an 11%p gap between the two settings in EM-User. Third, comparing MLP and GraphSAGE, GraphSAGE is better in FNTN, and MLP is better in EM-User. Leveraging the graph structure leads to better representation in FNTN, but not in EM-User.

Performance by the number of observed nodes In Figure 3, we show the mean performance of Intra/Inter-SGI (5 runs) by the number of observed nodes in the test and training. Here, we exclude HPO-Metab with an average number of nodes fewer than 64. Intuitively, more observations should result in better prediction, and the performance on EM-User is consistent with that intuition. However, for FNTN, the opposite is true because initial nodes are relatively important for the propagation-based fake news detection task (Bian et al., 2020). Note that adding observed nodes is equivalent to adding k-hop neighbors to be discriminated by $\mathcal{D}^{\text{obs}}$. That is, the impact of performance degradation from neighborhood noise is more significant than information gain from additional nodes in FNTN. See Appendix G for a more detailed discussion.

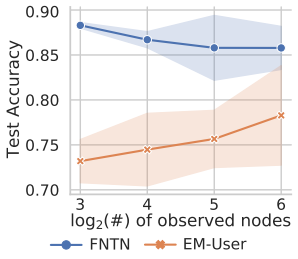


Figure 3: Performance by # of observed nodes at *test and training*. The shaded area is the standard deviation.

Generalization across sizes of *test* subgraphs We demonstrate how Intra/Inter-SGI generalizes across sizes of test subgraphs. In Figure 4, we report the mean performance (5 runs) by the number of observed nodes in the test stage. We set the number of *test* observed nodes from 4 to 64, and fix the number of *training* observed nodes to 8. Our model generalizes on test samples with more observed nodes (> 8) than training, but the variance of performances increases. In contrast, there is a lack of generalizability for test samples with fewer observed nodes than in the training stage. In particular, some cases do not converge depending on the initialization in EM-User.

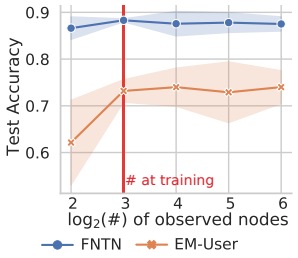


Figure 4: Performance by # of observed nodes at the *test*.

Performance by edge types in inclusive subgraphs In Table 2, we show the performance of Intra/Inter-SGI by edge types in $\boldsymbol{H}^{\pm} = \mathcal{E}^{\text{in}}(\boldsymbol{X}^{\pm}, \mathbb{A}^{\text{sub},\pm}$ or $\mathbb{A}^{\text{glob}}[\mathbb{V}^{\text{sub},\pm}])$ from the Inter-SGI loss. We use only FNTN since $\mathbb{A}^{\text{sub}}$ and $\mathbb{A}^{\text{glob}}[\mathbb{V}^{\text{sub}}]$ are the same in other datasets (§2.2, §4.1). The gap of 1.4%p is not significant (p-value of 0.09 in t-test), however, this difference suggests that when message-passing is performed through the edge induced in the global graph ($\mathbb{A}^{\text{glob}}[\mathbb{V}^{\text{sub}}]$), a more appropriate contrastive representation of nodes can be obtained for the InfoMax method.

Table 2: Performance by edge types on FNTN.

Edge type	FNTN
$\mathbb{A}^{\text{sub}}$	$88.2_{\pm 1.4}$
$\mathbb{A}^{\text{glob}}[\mathbb{V}^{\text{sub}}]$	$89.6_{\pm 1.0}$

6 Related Work

Subgraphs in graph representation learning There have been several approaches to use the information in subgraphs to improve representation learning of graph-structured data. They employ subgraphs to build more expressive models for node and graph representations (Niepert et al., 2016; Morris et al., 2019; Bouritsas et al., 2020; Peng et al., 2020a; Yu et al., 2021), improve the scalability of graph neural network (GNN) training (Hamilton et al., 2017; Chiang et al., 2019; Zeng et al., 2020b;a), augment data for graphs (Qiu et al., 2020; You et al., 2020), and explain prediction results of GNNs (Ying et al., 2019; Luo et al., 2020). Another common approach is to learn (or meta-learn) nodes or edges of interest by fetching a local (or enclosing) subgraph around them (Bordes et al., 2014; Zhang & Chen, 2018; Teru et al., 2020; Huang & Zitnik, 2020).

While these methods target node- or graph-level tasks, a few studies focus on the subgraph-level task. Meng et al. (2018) classifies the subgraph evolution pattern for subgraphs induced by three or four nodes as inputs, but it does not learn the representation of subgraphs. Subgraph Neural Network (SubGNN) (Alsentzer et al., 2020) is designed for subgraph-level classification with subgraph representation learning using their internal/external topology, positions, and connectivity. SubGNN assumes that all subgraphs are fully observed, whereas our model does not make this assumption.

Contrastive learning by local-global mutual information maximization Contrastive learning is a widely-used method for self- and un-supervised representation learning (Liu et al., 2020; Le-Khac et al., 2020). This has been applied in various types of data such as language (Mnih & Kavukcuoglu, 2013; Mikolov et al., 2013), nodes (Perozzi et al., 2014; Grover & Leskovec, 2016), and images (He et al., 2020; Chen et al., 2020). Within contrastive learning, local-global InfoMax methods (Hjelm et al., 2019) have been proposed recently, leveraging the known structure of data while maximizing mutual information (MI) of input and encoded output. Specifically, they maximize MI between pairs of local (e.g., patches) and global (e.g., images) based on neural MI estimators (Belghazi et al., 2018; Nowozin et al., 2016; Oord et al., 2018).

For graphs, various inherent substructures can be used in the design of contrastive learning. For example, node representations can be obtained by maximizing MI between node-graph pairs (Veličković et al., 2019; Park et al., 2020; Hassani & Khasahmadi, 2020; Wang et al., 2020; Jing et al., 2021), node-subgraph pairs (Peng et al., 2020b; Jiao et al., 2020; Li et al., 2020), edge-edge pairs (Peng et al., 2020b), and subgraph-graph pairs (Cao et al., 2021). Likewise, graph representations can be obtained by maximizing MI between node-graph pairs (Sun et al., 2020; Hassani & Khasahmadi, 2020), node-subgraph pairs (Li et al., 2020), and subgraph-graph pairs (Sun et al., 2021). Our model learns representations of partially observed subgraphs by maximizing MI between pairs of nodes and two different levels of subgraphs. To the best of our knowledge, there is no local-global InfoMax method designed to learn subgraph representation itself, regardless of the conditions of incomplete observations.

7 Conclusion

We explored the 'partial subgraph learning task' where only a part of the subgraph is observed. This is a more realistic and challenging scenario of subgraph representation learning. To solve this problem, we proposed a novel model, Intra- and Inter-Subgraph InfoMax (SGI), which maximizes the mutual information between the summary of the subgraph and node representations in it. Intra- and Inter-SGI sequentially summarizes the subgraph from a small observed set of nodes and larger k-hop neighborhoods, respectively, and performs contrastive optimization to reconstruct the true

subgraph representation. Based on training and evaluation protocols designed to simulate the real-world use cases, our experiments demonstrate that Intra- and Inter-SGI outperform baselines in three datasets across all settings. One limitation of this research is that Intra-SGI uses a naive k-hop sampling to select neighbors to be included in the subgraph, which is a major cause of performance degradation in dense graphs. Research on how to effectively and efficiently select nodes is necessary for this method, and we leave it as future work.

REFERENCES

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2623–2631, 2019.

Emily Alsentzer, Samuel G Finlayson, Michelle M Li, and Marinka Zitnik. Subgraph neural networks. *Proceedings of Neural Information Processing Systems, NeurIPS*, 2020.

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International Conference on Machine Learning*, pp. 531–540. PMLR, 2018.

Tian Bian, Xi Xiao, Tingyang Xu, Peilin Zhao, Wenbing Huang, Yu Rong, and Junzhou Huang. Rumor detection on social media with bi-directional graph convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 549–556, 2020.

Antoine Bordes, Sumit Chopra, and Jason Weston. Question answering with subgraph embeddings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 615–620, 2014.

Giorgos Bouritsas, Fabrizio Frasca, Stefanos Zafeiriou, and Michael M Bronstein. Improving graph neural network expressivity via subgraph isomorphism counting. *arXiv preprint arXiv:2006.09252*, 2020.

Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

Cătălina Cangea, Petar Veličković, Nikola Jovanović, Thomas Kipf, and Pietro Liò. Towards sparse hierarchical graph classifiers. *arXiv preprint arXiv:1811.01287*, 2018.

Jiangxia Cao, Xixun Lin, Shu Guo, Luchen Liu, Tingwen Liu, and Bin Wang. Bipartite graph embedding via mutual information maximization. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pp. 635–643, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Tong Chen, Xue Li, Hongzhi Yin, and Jun Zhang. Call attention to rumors: Deep attention based recurrent neural networks for early rumor detection. In *Pacific-Asia conference on knowledge discovery and data mining*, pp. 40–52. Springer, 2018.

Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 257–266, 2019.

Vijay Prakash Dwivedi, Chaitanya K Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *arXiv preprint arXiv:2003.00982*, 2020.

William Falcon and The PyTorch Lightning team. PyTorch Lightning, 3 2019. URL `https://github.com/PyTorchLightning/pytorch-lightning`.

Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *International Conference on Learning Representations Workshop on Representation Learning on Graphs and Manifolds*, 2019.

Hongyang Gao and Shuiwang Ji. Graph u-nets. In *international conference on machine learning*, pp. 2083–2092. PMLR, 2019.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 855–864, 2016.

William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 1025–1035, 2017.

Taila Hartley, Gabrielle Lemire, Kristin D Kernohan, Heather E Howley, David R Adams, and Kym M Boycott. New diagnostic approaches for undiagnosed rare genetic diseases. *Annual review of genomics and human genetics*, 21:351–372, 2020.

Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In *International Conference on Machine Learning*, pp. 4116–4126. PMLR, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bklr3j0cKX.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687*, 2020.

Kexin Huang and Marinka Zitnik. Graph meta learning via local subgraphs. *Advances in Neural Information Processing Systems*, 33, 2020.

Yizhu Jiao, Yun Xiong, Jiawei Zhang, Yao Zhang, Tianqi Zhang, and Yangyong Zhu. Sub-graph contrast for scalable self-supervised graph representation learning. *arXiv preprint arXiv:2009.10273*, 2020.

Baoyu Jing, Chanyoung Park, and Hanghang Tong. Hdmi: High-order deep multiplex infomax. *arXiv preprint arXiv:2102.07810*, 2021.

Jooyeon Kim, Dongkwan Kim, and Alice Oh. Homogeneity-based transmissive process to model true and false news in social networks. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pp. 348–356, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

Boris Knyazev, Graham W Taylor, and Mohamed Amer. Understanding attention and generalization in graph neural networks. *Advances in Neural Information Processing Systems*, 32:4202–4212, 2019.

Sebastian Köhler, Leigh Carmody, Nicole Vasilevsky, Julius O B Jacobsen, Daniel Danis, Jean-Philippe Gourdine, Michael Gargano, Nomi L Harris, Nicolas Matentzoglu, Julie A McMurry, et al. Expansion of the human phenotype ontology (hpo) knowledge base and resources. *Nucleic acids research*, 47(D1): D1018–D1027, 2019.

Phuc H Le-Khac, Graham Healy, and Alan F Smeaton. Contrastive representation learning: A framework and review. *IEEE Access*, 2020.

Maosen Li, Siheng Chen, Ya Zhang, and Ivor Tsang. Graph cross networks with vertex infomax pooling. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 14093–14105. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/a26398dca6f47b49876cbaffbc9954f9-Paper.pdf.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.

Xiao Liu, Fanjin Zhang, Zhenyu Hou, Zhaoyu Wang, Li Mian, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *arXiv preprint arXiv:2006.08218*, 1(2), 2020.

Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. Real-time rumor debunking on twitter. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pp. 1867–1870, 2015.

Yang Liu and Yi-Fang Wu. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. *Advances in Neural Information Processing Systems*, 33, 2020.

Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam-Fai Wong, and Meeyoung Cha. Detecting rumors from microblogs with recurrent neural networks.(2016). In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI 2016)*, pp. 3818–3824, 2016.

Jing Ma, Wei Gao, and Kam-Fai Wong. Detect rumors in microblog posts using propagation structure via kernel learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 708–717, 2017.

Jing Ma, Wei Gao, and Kam-Fai Wong. Rumor detection on twitter with tree-structured recursive neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1980–1989, 2018.

Changping Meng, S Chandra Mouli, Bruno Ribeiro, and Jennifer Neville. Subgraph pattern neural networks for high-order graph evolution prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 2*, pp. 3111–3119, 2013.

Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. *Advances in neural information processing systems*, 26:2265–2273, 2013.

Dylan Mordaunt, David Cox, and Maria Fuller. Metabolomics to improve the diagnostic efficiency of inborn errors of metabolism. *International journal of molecular sciences*, 21(4):1195, 2020.

Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4602–4609, 2019.

Taichi Murayama, Shoko Wakamiya, Eiji Aramaki, and Ryota Kobayashi. Modeling the spread of fake news on twitter. *Plos one*, 16(4):e0250419, 2021.

Jianmo Ni, Larry Muhlstein, and Julian McAuley. Modeling heart rate and activity data for personalized fitness recommendation. In *The World Wide Web Conference*, pp. 1343–1353, 2019.

Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *International conference on machine learning*, pp. 2014–2023. PMLR, 2016.

Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper/2016/file/cedebb6e872f539bef8c3f919874e9d7-Paper.pdf.

Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4004–4012, 2016.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Chanyoung Park, Donghyun Kim, Jiawei Han, and Hwanjo Yu. Unsupervised attributed multiplex network embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5371–5378, 2020.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pp. 8026–8037, 2019.

Hao Peng, Jianxin Li, Qiran Gong, Yuanxin Ning, Senzhang Wang, and Lifang He. Motif-matching based subgraph-level attentional convolutional network for graph classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5387–5394, 2020a.

Zhen Peng, Wenbing Huang, Minnan Luo, Qinghua Zheng, Yu Rong, Tingyang Xu, and Junzhou Huang. Graph representation learning via graphical mutual information maximization. In *Proceedings of The Web Conference 2020*, pp. 259–270, 2020b.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 701–710, 2014.

Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. Gcc: Graph contrastive coding for graph neural network pre-training. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1150–1160, 2020.

Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. In *International Conference on Learning Representations*, 2020.

Justin Sampson, Fred Morstatter, Liang Wu, and Huan Liu. Leveraging the implicit structure within social media for emergent rumor detection. In *Proceedings of the 25th ACM international on conference on information and knowledge management*, pp. 2377–2382, 2016.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823, 2015.

Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

Kai Shu, Deepak Mahudeswaran, Suhang Wang, and Huan Liu. Hierarchical propagation networks for fake news detection: Investigation and exploitation. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 14, pp. 626–637, 2020.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56): 1929–1958, 2014. URL http://jmlr.org/papers/v15/srivastava14a.html.

Fan-Yun Sun, Jordan Hoffman, Vikas Verma, and Jian Tang. Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1lfF2NYvH.

Qingyun Sun, Jianxin Li, Hao Peng, Jia Wu, Yuanxing Ning, Phillip S. Yu, and Lifang He. Sugar: Subgraph neural network with reinforcement pooling and self-supervised mutual information mechanism, 2021.

Komal Teru, Etienne Denis, and Will Hamilton. Inductive relation prediction by subgraph reasoning. In *International Conference on Machine Learning*, pp. 9448–9457. PMLR, 2020.

Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6827–6839. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/4c2e5eaae9152079b9e95845750bb9ab-Paper.pdf.

Michael Tschannen, Josip Djolonga, Paul K. Rubenstein, Sylvain Gelly, and Mario Lucic. On mutual information maximization for representation learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkxoh24FPH.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30: 5998–6008, 2017.

Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rklz9iAcKQ.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103, 2008.

Pengyang Wang, Yanjie Fu, Yuanchun Zhou, Kunpeng Liu, Xiaolin Li, and Kien Hua. Exploiting mutual information for substructure-aware graph representation learning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pp. 3415–3421, 2020.

Mike Wu, Milan Mosse, Chengxu Zhuang, Daniel Yamins, and Noah Goodman. Conditional negative sampling for contrastive learning of visual representations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=v8b3e5jN66j.

Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems*, 32:9240, 2019.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33, 2020.

Junchi Yu, Tingyang Xu, Yu Rong, Yatao Bian, Junzhou Huang, and Ran He. Graph information bottleneck for subgraph recognition. In *International Conference on Learning Representations (ICLR)*, 2021.

Hanqing Zeng, Muhan Zhang, Yinglong Xia, Ajitesh Srivastava, Andrey Malevich, Rajgopal Kannan, Viktor Prasanna, Long Jin, and Ren Chen. Deep graph neural networks with shallow subgraph samplers. *arXiv preprint arXiv:2012.01380*, 2020a.

Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Graphsaint: Graph sampling based inductive learning method. In *International Conference on Learning Representations*, 2020b. URL https://openreview.net/forum?id=BJe8pkHFwS.

Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 5171–5181, 2018.

Chengxu Zhuang, Alex Lin Zhai, and Daniel Yamins. Local aggregation for unsupervised learning of visual embeddings. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6002–6012, 2019.

A NOTATION

We mostly followed notations from Goodfellow et al. (2016) provided by ICLR formatting instruction. We summarize our specific notations in Table 3.

Table 3: Meanings of the notations

Notation	Meaning
$\mathcal{G} = (\mathbb{V}^{\text{glob}}, \mathbb{A}^{\text{glob}}, \boldsymbol{X}^{\text{glob}})$	A global graph
$\mathbb{S} = \{\mathcal{S}_1, ..., \mathcal{S}_M\},\ \mathbb{S}^{\text{obs}} = \{\mathcal{S}_1^{\text{obs}}, ..., \mathcal{S}_M^{\text{obs}}\}$	Sets of M subgraphs and partially observed subgraphs
$\mathcal{S} = (\mathbb{V}^{\text{sub}}, \mathbb{A}^{\text{sub}}),\ \mathcal{S}^{\text{obs}} = (\mathbb{V}^{\text{obs}}, \mathbb{A}^{\text{obs}})$	A subgraph and a partially observed subgraph
$\mathbb{V}^{\text{glob}}, \mathbb{V}^{\text{sub}}, \mathbb{V}^{\text{obs}}$	Sets of nodes in $\mathcal{G}$, $\mathcal{S}$, $\mathcal{S}^{\text{obs}}$
$\mathbb{A}^{\text{glob}}, \mathbb{A}^{\text{sub}}, \mathbb{A}^{\text{obs}}$	Sets of edges in $\mathcal{G}$, $\mathcal{S}$, $\mathcal{S}^{\text{obs}}$
$\boldsymbol{X}^{\text{glob}} \in \mathbb{R}^{\lvert\mathbb{V}^{\text{glob}}\rvert \times F^{\text{in}}}$	An input feature matrix of global nodes $\mathbb{V}^{\text{glob}}$
$\boldsymbol{g}$	A vector of subgraph-level features
F^{in}, F	Dimensions of $\boldsymbol{X}^{\text{glob}}$ and the learned representation
$y \in \{1, ..., C\}$	A label (class) corresponding to the subgraph $\mathcal{S}$
$\mathcal{G}[\mathbb{V}^{\text{sub}}] = (\mathbb{V}^{\text{sub}}, \mathbb{A}[\mathbb{V}^{\text{sub}}])$	An induced subgraph of $\mathcal{G}$ formed from a set of nodes in the subgraph $\mathbb{V}^{\text{sub}}$
$\mathbb{V}^{\text{sub}_k}, \mathbb{V}^{\text{glob}_k}$	The k-hop neighbors of observed nodes $u \in \mathbb{V}^{\text{obs}}$ that are actually included in the subgraph $\mathbb{V}^{\text{sub}}$, and that are not.
$\mathbb{V}^+, \mathbb{V}^-$	Positive and negative sets of nodes for Inter-SGI
$\mathcal{E}^{\text{in}} : \mathbb{R}^{N \times F^{\text{in}}} \times \mathbb{A} \to \mathbb{R}^{N \times F}$	A graph neural encoder for input features $\boldsymbol{X}^{\text{glob}}, \mathbb{A}^{\text{glob}}$
$\mathcal{E}^Q, \mathcal{E}^K, \mathcal{E}^V, \mathcal{E}^I : \mathbb{R}^{N \times F} \to \mathbb{R}^{N \times F}$	Encoders for Intra-SGI (Q, K, V) and Inter-SGI (I)
$\boldsymbol{H}^{\text{obs}}, \boldsymbol{H}^{\text{sub}_k}, \boldsymbol{H}^{\text{glob}_k}$ and $\boldsymbol{H}^+, \boldsymbol{H}^-$	Encoded node representations of $\mathbb{V}^{\text{obs}}, \mathbb{V}^{\text{sub}_k}, \mathbb{V}^{\text{glob}_k}$ (Intra-SGI) and $\mathbb{V}^+, \mathbb{V}^-$ (Inter-SGI)
$\boldsymbol{H}^*, \boldsymbol{H}^{\pm}$	Concatenation of $\boldsymbol{H}^{\text{obs}}, \boldsymbol{H}^{\text{sub}_k}, \boldsymbol{H}^{\text{glob}_k}$ (Intra-SGI) and $\boldsymbol{H}^+, \boldsymbol{H}^-$ (Inter-SGI)
$\boldsymbol{H}^{Q,\text{obs}}, \boldsymbol{H}^{K,*}, \boldsymbol{H}^{V,*}, \boldsymbol{H}^{I,\pm}$	The output of $\mathcal{E}^Q$, $\mathcal{E}^K$, $\mathcal{E}^V$, $\mathcal{E}^I$ on $\boldsymbol{H}^{\text{obs}}$, $\boldsymbol{H}^*$, $\boldsymbol{H}^*$, $\boldsymbol{H}^{\pm}$
$\boldsymbol{s}^{\text{obs}}, \boldsymbol{s}^{\text{sub}} \in \mathbb{R}^F$	The subgraph summary vectors of node representations $\boldsymbol{H}^{Q,\text{obs}}, \boldsymbol{H}^{V,*}$ in subgraphs $\mathcal{S}^{\text{obs}}, \mathcal{S}$
$\mathcal{R}^{\text{obs}}, \mathcal{D}^{\text{obs}}$ and $\mathcal{R}^{\text{sub}}, \mathcal{D}^{\text{sub}}$	Readout functions and discriminators for Intra-SGI ($\boldsymbol{H}^{\text{obs}}, \boldsymbol{s}^{\text{obs}}$) and Inter-SGI ($\boldsymbol{H}^*, \boldsymbol{s}^{\text{sub}}$)

B PROOF OF PROPOSITION 1

Proposition 1 (The conditional GAN-like divergence MI bound)**.** *For d-dimensional random variables X and Y with a joint distribution $p(x, y)$ and marginal distributions $p(x)$ and $p(y)$, fix any function $f : (X, Y) \to \mathbb{R}$ and realization x of X. Let $c_x = \mathbb{E}_{y \sim p(y)}\left[e^{f(x,y)}\right]$, $\mathbb{B}_{c_x} \subset \mathbb{R}$ be strictly lower bounded by c_x, and $\mathbb{Y}_{c_x} = \{y | e^{f(x,y)} \in \mathbb{B}_{c_x}\}$ with an assumption of $p(\mathbb{Y}_{c_x}) > 0$. For $\mathbb{Y}_r$ in the Borel σ-algebra over $\mathbb{R}^d$, let $q(Y \in \mathbb{Y}_r | X = x) = p(\mathbb{Y}_r | \mathbb{Y}_{c_x})$, then $\mathcal{I}^{CGAN} \leq \mathcal{I}^{GAN}$ where*

$$\mathcal{I}^{CGAN} = \mathbb{E}_{x,y \sim p(x,y)}\left[\log \sigma(f(x, y))\right] + \mathbb{E}_{x \sim p(x)} \mathbb{E}_{y \sim q(y|x)}\left[\log\left(1 - \sigma(f(x, y))\right)\right], \tag{14}$$

$$\mathcal{I}^{GAN} = \mathbb{E}_{x,y \sim p(x,y)}\left[\log \sigma(f(x, y))\right] + \mathbb{E}_{x \sim p(x)} \mathbb{E}_{y \sim p(y)}\left[\log\left(1 - \sigma(f(x, y))\right)\right]. \tag{15}$$

Proof. It suffices to show that $\mathbb{E}_{y \sim p(y)}\left[\log\left(1 + e^{f(x,y)}\right)\right] \leq \mathbb{E}_{y \sim q(y|x)}\left[\log\left(1 + e^{f(x,y)}\right)\right]$ for all x to prove $\mathcal{I}^{\text{CGAN}} \leq \mathcal{I}^{\text{GAN}}$, since,

$$\mathbb{E}_{y \sim p(y)}\left[\log\left(1 + e^{f(x,y)}\right)\right] \leq \mathbb{E}_{y \sim q(y|x)}\left[\log\left(1 + e^{f(x,y)}\right)\right] \tag{16}$$

$$\Rightarrow \mathbb{E}_{x \sim p(x), y \sim p(y)}\left[\log\left(1 + e^{f(x,y)}\right)\right] \leq \mathbb{E}_{x \sim p(x), y \sim q(y|x)}\left[\log\left(1 + e^{f(x,y)}\right)\right] \tag{17}$$

$$\Rightarrow \mathbb{E}_{x \sim p(x), y \sim q(y|x)}\left[\log\left(1 - \sigma(f(x, y))\right)\right] \leq \mathbb{E}_{x \sim p(x), y \sim p(y)}\left[\log\left(1 - \sigma(f(x, y))\right)\right] \tag{18}$$

$$\Rightarrow \mathcal{I}^{\text{CGAN}} \leq \mathcal{I}^{\text{GAN}} \tag{19}$$

We apply the similar technique in CNCE (Wu et al., 2021) to prove Equation 16. Using Jensen's inequality to the right-hand side and the fact that $\mathbb{E}_p(e^{f(x,y)}) \leq e^{f(x,y_c)}$ for $y_c \in \mathbb{Y}_c$,

$$\mathbb{E}_p\left[\log\left(1+e^{f(x,y)}\right)\right] \leq \log \mathbb{E}_p\left[\left(1+e^{f(x,y)}\right)\right] \leq \log\left(1+e^{f(x,y_c)}\right). \tag{20}$$

If we take the expectation $\mathbb{E}_{y\sim q(y|x)}$ on both sides, we get Equation 16. □

C Dataset and Pre-processing Details

C.1 Dataset

In Table 4, we summarize the statistics of three real-world datasets: FNTN, EM-User, and HPO-Metab. Next paragraphs describe the components of these datasets (e.g., $\mathcal{G}$ and $\mathbb{S}$) and what tasks correspond to these datasets.

FNTN for fake news early detection (ordered, inclusive) FNTN (Fake News in Twitter Network) dataset consists of a follower network on Twitter ($\mathcal{G}$), propagation trees of news ($\mathbb{S}$), and TF-IDF vectors of its contents ($\boldsymbol{g}$). Taking FNTN dataset from prior studies (Liu et al., 2015; Ma et al., 2016; 2017; 2018; Kim et al., 2019), we add the follower network of users who propagated news retrieving them with the Twitter API. Fake news early detection is a classification task of the genuineness of news articles by the early propagated nodes (i.e., users who initially spread the news) (Sampson et al., 2016; Liu & Wu, 2018; Chen et al., 2018; Shu et al., 2020; Bian et al., 2020).

EM-User for user profiling with partial observation (unordered, identical) EM-User (Users in EndoMondo) dataset (Alsentzer et al., 2020) consists of a social fitness network $\mathcal{G}$ from Endomondo (Ni et al., 2019) and subgraphs $\mathbb{S}$ corresponding to users. Nodes are workouts, and edges exist between workouts completed by multiple users. Each subgraph $\mathcal{S}$ represents a user's workout history. The user profiling task with partial observation is to predict a user's gender with only a few sampled logs.

HPO-Metab for disease diagnosis with partial observation (unordered, identical) The global graph $\mathcal{G}$ of HPO-Metab (Metabolic disease in Human Phenotype Ontology) dataset (Alsentzer et al., 2020; Hartley et al., 2020; Köhler et al., 2019; Mordaunt et al., 2020) is a knowledge graph of phenotypes (i.e., symptoms) of rare diseases. Each subgraph $\mathcal{S}$ is a collection of symptoms associated with a monogenic metabolic disorder. Our task is to diagnose the rare disease: to classify the disease type among six subcategories of the metabolic disorders, assuming only some of the symptoms have been observed.

Split We randomly split the train/val/test set of FNTN with a ratio of 70/15/15 and use the public split (Alsentzer et al., 2020) for EM-User (70/15/15) and HPO-Metab (80/10/10).

Download The raw datasets of HPO-Metab and EM-User can be downloaded from SubGNN's GitHub repository[1]. We put the pre-processed FNTN dataset in the supplementary material.

C.2 Pre-processing

For FNTN, a follower network was crawled through the Twitter API between October and November 2018 for users in the propagation trees (including leaf users) (Liu et al., 2015; Ma et al., 2016; 2017; 2018; Kim et al., 2019). For deactivated accounts, we reflect the following information that can be obtained from the tree. We collect and distribute these data under Twitter's policies and agreements[2].

Datasets in this paper are pre-processed to remove any personally identifiable information of users in real-world services (Twitter for FNTN and Endomondo for EM-User). Users are fully anonymized and treated as consecutive integers. In addition, we take TF-IDF vectors of 2000 words for news content without stop-words. The fake news texts, which can be offensive, cannot be restored.

For all datasets, single node graphs are excluded (five subgraphs for EM-User and three subgraphs for HPO-Metab). The rests are the same as the original papers (See Kim et al. (2019) for FNTN, and Alsentzer et al. (2020) for EM-User and HPO-Metab).

[1] https://github.com/mims-harvard/SubGNN

[2] https://developer.twitter.com/en/developer-terms

Table 4: Statistics of real-world datasets.

	FNTN	EM-User	HPO-Metab
# Global nodes	362232	57333	14587
# Global edges	22918295	4573417	3238174
Density of the global graph	0.0002	0.0028	0.0304
Whether edges are directed	directed	undirected	undirected
# Nodes per subgraph	408.61 ± 386.72	155.42 ± 100.38	14.44 ± 6.19
# Edges per subgraph	412.92 ± 391.32	534.86 ± 645.30	181.30 ± 181.83
Density of subgraphs	0.0043 ± 0.0027	0.0159 ± 0.0052	0.7576 ± 0.1486
# Subgraphs	1107	319	2397
Split	Random	Public	Public
Train/Val/Test	775/166/166	224/48/49	1918/244/235
# Classes	4	2	6

D Model, Training, and Hyperparameter Configurations

D.1 Model and Training Details

In addition to the description in the main paper, we use the following model architectures and training methods:

- All the models are implemented with PyTorch (Paszke et al., 2019), PyTorch Geometric (Fey & Lenssen, 2019), and PyTorch Lightning (Falcon & The PyTorch Lightning team, 2019).
- When using a bidirectional encoder, half of the hidden feature of 64 is divided and used for each direction. That is, we use 32 for forward edges and 32 for reverse edges.
- For positional encoding, we follow the Transformer's original formula (Vaswani et al., 2017) and set the maximum length of 20. Note that the number of observed nodes is 8. When the numbers of observed nodes are 16, 32, and 64, the maximum lengths are 36, 68, and 132, respectively.
- A fixed number (N^{obs}) of observed nodes is sampled at each iteration of the training stage. To add more randomness, we sample a random element from $\{N^{\text{obs}} - 2, N^{\text{obs}} - 1, N^{\text{obs}}, N^{\text{obs}} + 1, N^{\text{obs}} + 2\}$ first and select the observed nodes of that number.
- For the k-hop subgraph, we dropout these edges with the probability of p_d (Rong et al., 2020).
- The batch sizes are 16 for FNTN and 64 for others, using the gradient accumulation (16 for FNTN, 1 for EM-User, and 16 for HPO-Metab).
- Intra-SGI uses the same number of negative samples as the number of positive samples belonging to the k-hop subgraph. Inter-SGI uses the entire nodes in the negative subgraph.
- All model parameters are trained with 16-bit precision supported by PyTorch Lightning (Falcon & The PyTorch Lightning team, 2019).
- Each of our experiments is done on a single GPU. These GPUs are the GeForce GTX 1080 Ti, GeForce RTX 2080 Ti, and Quadro RTX 8000, but each experiment does not require a specific GPU type. One machine has a total of 40 – 48 cores of CPUs and 4 – 8 GPUs.

D.2 Hyperparameter Selection

We tune a subset of hyperparameters with validation sets using Akiba et al. (2019). Depending on models and experiment conditions, we choose different tuning algorithms and hyperparameter subsets. For the MLP and GraphSAGE models, only weight decay is tuned. For SubGNN, we compare the model with all (neighborhood, structure, and position) channels and models with only one channel each. For each case, we tune weight decay, an aggregator for initializing component embedding, k for k-hop neighborhood of subgraph component, and numbers of structure anchor patches, border/internal position anchor patches, border/internal neighborhood anchor patches, and LSTM layers for structure anchor patch embedding. Lastly, we tune weight decay, λ^{intra}, λ^{inter}, ratio in top_k pooling, and DropEdge probability p_d of k-hop subgraph for our models.

Table 5: Summary of accuracy (5 runs) of Intra/Inter-SGI with and without using Transformer in the encoder $\mathcal{E}^Q$ (Equation 10). The result of GraphSAGE in the best setting is also presented.

Model	FNTN	EM-User
GraphSAGE (Best)	85.9 ± 1.0	68.5 ± 3.2
Intra/Inter-SGI with Transformer	89.6 ± 2.2	77.0 ± 2.8
Intra/Inter-SGI without Transformer	87.8 ± 0.8	74.5 ± 4.5

We use the Tree-structured Parzen Estimator (TPE) algorithm under a total budget of 50 runs for most experiments. For SubGNN, we choose the random search following the original implementation. Exceptionally, we perform the grid search for a more controlled evaluation of Intra/Inter-SGI according to the number of observed nodes in FNTN and EM-User datasets. We run a total of 36 experiments on the space of three λ^{intra} ($\{1.0, 2.0, 3.0\}$), three λ^{inter} ($\{1.0, 2.0, 3.0\}$), two weight decay ($\{10^{-4}, 10^{-3}\}$ for FNTN and $\{10^{-6}, 10^{-5}\}$ for others), and two pool ratio ($\{10^{-4}, 10^{-3}\}$ for HPO-Metab and $\{10^{-3}, 10^{-2}\}$ for others).

All hyperparameters are reported in `./SGI/args.yaml` in the code.

E Ablation Study and Hyperparameter Sensitivity Analysis

E.1 Ablation Study on Model Architectures

We conduct ablation experiments on the Transformer in the encoder $\mathcal{E}^Q$ (Equation 10) on FNTN and EM-User datasets. In this study, we replace the Transformer in $\mathcal{E}^Q$ with the two-layer MLP. For MLPs, we do not use the positional encoding on both datasets.

In Table 5, we summarize the mean accuracy over five runs of GraphSAGE and Intra/Inter-SGI. We reprint the result of GraphSAGE model in the best setting for each dataset. Using the Transformer in $\mathcal{E}^Q$ improves the performance by 1.8%p (FNTN) and 2.5%p (EM-User). This result shows that modeling pairwise interactions between observed nodes contributes to learning fine representation of subgraphs. However, even without Transformer, our model outperforms the GraphSAGE models.

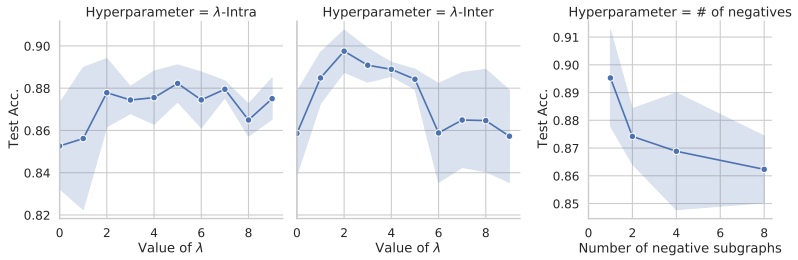
E.2 Hyperparameter Sensitivity Analysis

We analyze how the performance varies according to λ^{intra}, λ^{inter}, and the number of negative samples for the Inter-SGI loss (i.e., $|\mathbb{V}^-|$). In Figure 5, we draw the plot of test accuracy on FNTN and EM-User against these three hyperparameters. As with other experiments, five experiments are conducted, and the average performance is reported with standard deviation as shaded area. We employ the Intra-SGI for λ^{intra} and the Intra/Inter-SGI for λ^{inter} and the number of negative samples. For the number of negative samples, since we use nodes in an individual subgraph as negatives in the Inter-SGI part, we report the performance according to the number of negative *subgraphs*.

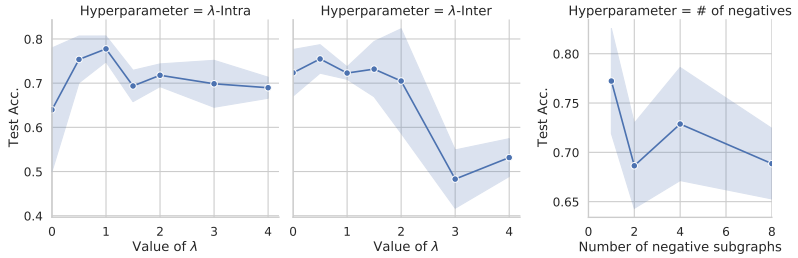
We have three observations in the analysis of λ^{intra} and λ^{inter}. First, using Intra-SGI and Inter-SGI loss ($\lambda > 0$) contributes to performance improvements. Second, the standard deviation increases in a range outside the optimal value. Lastly, the sensitivity for λ^{inter} is higher than λ^{intra} in both datasets.

Notably, in the region where λ^{intra} is larger than the optimum, the performance fluctuates slightly less. A larger λ^{intra} means that we train the discriminator $\mathcal{D}^{\text{obs}}$ to overestimate the probability of belonging to the subgraph. Since we only use a fixed ratio in the top_k pooling, those classified as nodes belonging to the subgraph more than this ratio do not involve in the subgraph representation for actual classification. This is why there is no significant change in the region of large λ^{intra}.

For the number of negative samples in the Inter-SGI loss, the ablation study demonstrates that increasing the number of negative subgraphs more than one hurts the performance. For both datasets, the decrements of performance is the largest when the number of negatives is raised from 1 to 2. The performance difference between 2 – 8 cases is slight.



(a) FNTN



(b) EM-User

Figure 5: Mean test accuracy over five runs on FNTN and EM-User against λ^{intra}, λ^{inter} and the number of negative subgraphs for the Inter-SGI loss. The shaded area represents the standard deviation.

Table 6: Comparison of k-hop subgraphs and full subgraphs for different graph characteristics: degree assortativity, average clustering, and density.

Dataset	Graph characteristic	k-hop subgraphs	Full subgraphs
FNTN	Degree assortativity	-0.213 ± 0.053	-0.483 ± 0.093
	Average clustering	0.370 ± 0.061	0.140 ± 0.136
	Density	0.005 ± 0.010	0.010 ± 0.015
EM-User	Degree assortativity	-0.097 ± 0.015	-0.109 ± 0.084
	Average clustering	0.066 ± 0.009	0.022 ± 0.011
	Density	0.017 ± 0.003	0.016 ± 0.002
HPO-Metab	Degree assortativity	-0.277 ± 0.028	-0.281 ± 0.107
	Average clustering	0.731 ± 0.016	0.874 ± 0.089
	Density	0.211 ± 0.095	0.787 ± 0.126

F Comparison Between k-hop and Full Subgraphs

We hypothesize that we can see the benefits of both models when the k-hop subgraph (for Intra-SGI) and the full subgraph (for Inter-SGI) provide different views in contrastive learning. We compare several graph properties of k-hop and full subgraphs to investigate whether the differences in properties are related to performance improvement by concurrent use of SGIs. We measure degree assortativity, average clustering, and density, and report the mean and standard deviation of 50 random samples in Table 6. We select these properties because they are well-defined for the individual subgraph, regardless of whether an edge has a direction or whether a subgraph is connected.

We confirm that for FNTN, the difference between k-hop and full subgraphs is twice or more in all properties. However, in the case of EM-User, there is no significant difference in degree assortativity and density. This is consistent with our assumption that differences in subgraph characteristics lead to differences in views and eventually provide different learning gains of contrast. For HPO-Metab, there is an extreme difference in density, but the performance degradation from the size of the k-hop subgraph in Intra-SGI seems to have a greater impact on performance change. (See §5 about this discussion).

Table 7: Summary of accuracy (5 runs) of GraphSAGE model on three datasets with regard to the ratio of observed nodes at the test and training (i.e., $x\%/x\%$ setting).

The ratio of observed nodes	FNTN	EM-User	HPO-Metab
12.5%	85.9 ± 1.3	54.5 ± 19.4	34.2 ± 2.1
25%	86.3 ± 0.7	82.6 ± 3.5	41.2 ± 1.3
100%	86.3 ± 0.7	82.1 ± 1.2	47.7 ± 3.3

Table 8: Mean wall-clock time (seconds) per batch of the training process on real-world datasets.

Model	FNTN	EM-User	HPO-Metab
MLP	0.021	0.040	0.018
GraphSAGE	0.028	0.037	0.019
SubGNN	N/A	0.126	0.086
Intra-SGI	0.816	0.103	0.406
Inter-SGI	0.047	0.053	0.033
Intra/Inter-SGI	0.834	0.141	0.413

G Discussion on Performance by the Number of Observed Nodes

In Figure 3, we show the performance of Intra/Inter-SGI depending on the number of observed nodes. By observing more nodes, the performance on EM-User increases but that on FNTN decreases. We claim that the impact of performance degradation from neighborhood noise is more significant than information gain from additional nodes in FNTN.

Where the boundary of the full subgraph is unknown, the increase in the number of observed nodes presents the following two challenges. First, the number of nodes that belong to the sampled k-hop neighborhood increases. Second, the number of nodes that are in the subgraph but unknown yet to the model decreases. We expect that the performance increases when the information gain from the additional nodes exceeds the noises from the above challenges.

Next, we show that initial nodes are relatively important for FNTN dataset. In Table 7, we report the experimental result of the GraphSAGE model with regard to the ratio of observed nodes. We call this the $x\%/x\%$ setting similar to the 100%/100% setting but uses only $x\%$ of nodes in training and evaluation. We set x to 12.5, 25, and 100. As the number of observed nodes decreases, the performance of the GraphSAGE model for all datasets generally decreases. However, the degree varies by dataset. Compared to EM-User and HPO-Metab, additional observed nodes in FNTN do not significantly affect subgraph representation quality. This is in line with Bian et al. (2020).

We demonstrate that the information gain from additional nodes in FNTN is relatively small. Considering the challenges from these additions, this explains why the performance of SGI on FNTN decreases as the number of observed nodes increases in Figure 3.

H Efficiency Comparison on Time and Memory

H.1 Training Time by Models and Datasets

In Table 8, we report the mean wall-clock time per batch using a single machine (40-core CPU with one GTX1080Ti GPU). For all experiments, we use a batch size of four.

The Inter-SGI does not show much difference with the baseline GraphSAGE and MLP in training time. The overhead is below 0.03s for all datasets. However, the model using Intra-SGI shows a relatively large training time compared to others ($\times 3 - \times 30$). We confirm that most of the increments occur from k-hop sampling. The more edges (e.g., FNTN) or density (e.g., HPO-Metab) of the global graph, the more time it takes to run. The EM-User dataset with a relatively low value for these properties takes a similar training time to that of SubGNN.

Table 9: The number of parameters of GraphSAGE and Intra/Inter-SGI.

Model	FNTN	EM-User	HPO-Metab
GraphSAGE	11736644	3694274	958790
Intra/Inter-SGI	11816070	3706690	971206
Rate of increment (%)	0.68	0.34	1.29

H.2 Number of Parameters

In Table 9, we compare the total number of parameters of GraphSAGE and Intra/Inter-SGI. This parameter includes not only the model but also the learnable node embeddings. As we can see in the last row, the number of parameters that our model uses more than the GraphSAGE model is around 1%, which we can consider negligible.

I Ethical Considerations

Learning subgraphs requires collecting more attributes (i.e., a global graph plus subgraphs) than learning nodes, edges, and graphs. This could lead to privacy invasion depending on the use case. For example, if we set the global graph as a user network of a social media like FNTN and EM-User, our model should follow up the entire network throughout its life cycle of training and evaluation.

Furthermore, a deeper understanding of the partial subgraph learning problem may enable harmful applications, such as tracking users on social media. Indeed, our study deals with the profiling task of users' gender (EM-User). Similar concerns are raised in the SubGNN paper, which proposed the original dataset (See Broader Impact section in Alsentzer et al. (2020)). Also, while our model suggests the positive application of fake news detection, but it leaves room for attacks to deceive. This is a general problem with any machine learning model, and thus researchers accessing and using this research must be mindful of potential harm.

Lastly, EM-User dataset simulates the prediction task of binary genders (male and female), but genders could be non-binary in reality. Future research should consider that EM-User is an oversimplified dataset for the benchmarking purpose.

J Figure Attribution

The Figure 1 is created with slightly modified 'User' icon by Made x Made Icons, 'Cough' icon by monkik, 'sleepless' icon by Andrejs Kirma, 'vomit' icon by Mini Hong, 'Weight Loss' icon by counloucon, 'Liver' icon by Lagot Design, 'lung disease' icon by Dooder, 'kidney disease' icon by Llisole, 'Disease' icon by WEBTECHOPS LLP, and 'fever' icon by popcornarts from the Noun Project.