Models and Benchmarks for Representation Learning of Partially Observed Subgraphs

Dongkwan Kim
dongkwan.kim@kaist.ac.kr
Korea Advanced Institute of Science and Technology
Daejeon, Republic of Korea

Jaimeen Ahn
jaimeen01@kaist.ac.kr
Korea Advanced Institute of Science and Technology
Daejeon, Republic of Korea

Alice Oh
alice.oh@kaist.edu
Korea Advanced Institute of Science and Technology
Daejeon, Republic of Korea

ABSTRACT

Subgraphs are rich substructures in graphs, and their nodes and edges can be partially observed in real-world tasks. Under partial observation, existing node- or subgraph-level message-passing produces suboptimal representations. In this paper, we formulate a novel task of learning representations of partially observed subgraphs. To solve this problem, we propose Partial Subgraph InfoMax (PSI) framework and generalize existing InfoMax models, including DGI, InfoGraph, MVGRL, and GraphCL, into our framework. These models maximize the mutual information between the partial subgraph’s summary and various substrutures from nodes to full subgraphs. In addition, we suggest a novel two-stage model with $k$-hop PSI, which reconstructs the representation of the full subgraph and improves its expressiveness from different local-global structures. Under training and evaluation protocols designed for this problem, we conduct experiments on three real-world datasets and demonstrate that PSI models outperform baselines.

KEYWORDS

graph neural networks; subgraph representation learning; mutual information maximization

CCS CONCEPTS

• Computing methodologies → Learning latent representations; Neural networks; Supervised learning by classification.

1 INTRODUCTION

The graph neural network (GNN) has become a major framework to learn representations of nodes, edges, and graphs [2, 5, 6, 14]. In addition, subgraphs can express various real-world data: news propagation in a social network or disease in a graph of symptoms [1]. The current formulation of subgraph representation learning by Alsentzer et al. [1] assumes full observation of nodes and edges in a subgraph, and that assumption often does not hold in the real world. If so, existing models may learn suboptimal representations because of inaccurate message-passing with missing nodes or edges. In this paper, we suggest a novel task of learning representations of partial subgraphs by relaxing the assumption of complete observation.

For this ‘partial subgraph learning’ task, we propose the Partial Subgraph InfoMax (PSI) framework based on mutual information (MI) maximization. Inspired by Deep InfoMax [13] that maximizes MI between the global summary (e.g., images) and local parts (e.g., patches), PSI maximizes MI between a partial subgraph and its substructures (e.g., nodes or full subgraphs). We generalize existing InfoMax models for node and graph-level tasks [11, 30, 33, 34] to solve the partial subgraph learning problem. PSI models first summarize a specific partial subgraph and learn to distinguish for its summary whether a substructure is related to the same subgraph (positive) or not (negative). This allows learning the structural hierarchy of nodes, partial and full subgraphs in subgraph representations.

However, the summary of the partial subgraph cannot explicitly encode missing information. Thus, we employ two-stage PSI models that reconstruct the summary close to the full subgraph under insufficient observation. We first propose ‘$k$-hop PSI’ that reconstructs the full subgraph by assembling $k$-hop neighbors of high MI with the partial subgraph. Then, the second PSI model takes the reconstructed subgraph summary as input and learns local-global structures different from the first $k$-hop PSI.

We demonstrate the improved representation learning performance of PSI models with experiments on three real-world datasets. These datasets simulate scenarios of fake news early detection, social network user profiling, and disease diagnosis with partial observation (Figure 1). Our models consistently outperform baseline models for all datasets. In addition, we analyze models’ performance depending on the properties of subgraphs and the global graph.

We present the following contributions. First, we formulate the partial subgraph learning problem and suggest realistic training and evaluation protocols (§2). Second, we propose the Partial Subgraph
Figure 1: Left: Comparison of subgraph and partial subgraph representation learning. Middle & Right: Real-world examples (§2) of partial subgraph learning.

2 PARTIAL SUBGRAPH LEARNING PROBLEM

We formulate a novel problem of learning subgraph representations under partial observations of nodes and edges.

Notations. Let $G = (V^{\text{glob}}, A^{\text{glob}})$ be a global graph, where $V^{\text{glob}}$ is a set of nodes and $A^{\text{glob}}$ is a set of edges, and $X^{\text{glob}} \in \mathbb{R}^{|V^{\text{glob}}| \times |E^{\text{glob}}|}$ is a feature matrix of nodes. A subgraph $S = (V^{\text{sub}}, A^{\text{sub}})$ of $G$ is defined as a graph, the nodes and edges of which are subsets of $V^{\text{glob}}$ and $A^{\text{glob}}$, respectively. Each subgraph has a label $y \in \{1, ..., C\}$, and sometimes, a subgraph-level feature $\bar{g} \in \mathbb{R}^F$ may exist.

Problem formulation. We formulate the ‘partial subgraph learning’ by relaxing the complete observation assumption of Alsentzer et al. [1], considering a subset of nodes or edges of the subgraph, as in Figure 1. We define a partial subgraph $S^{\text{obs}}$ of $S$ as $S^{\text{obs}} = (V^{\text{obs}}, A^{\text{obs}})$, where $V^{\text{obs}} \subseteq V^{\text{sub}}$ and $A^{\text{obs}} \subseteq A^{\text{sub}}$. We denote a set of subgraphs as $\mathcal{S} = \{S_1, ..., S_M\}$, and corresponding partial subgraphs as $\mathcal{S}^{\text{obs}} = \{S_1^{\text{obs}}, ..., S_M^{\text{obs}}\}$, where each $S_i^{\text{obs}}$ is a subgraph of $S_i$ for $i \in \{1, ..., M\}$. We aim to learn a representation $\bar{s} \in \mathbb{R}^F$ for each $S_i^{\text{obs}} \in \mathcal{S}^{\text{obs}}$ to predict $y$.

Real-world examples. One example of a real-world scenario is in detection of fake news in a social network, where the propagation tree of news can be represented as a subgraph. Rather than a fully propagated subgraph, it is more useful to detect fake news with an early propagated subgraph before the news spreads. Here, nodes are observed according to the order in which the information is propagated, and a partial subgraph would contain only nodes of the early propagation. Another example is a noisy diagnosis task of diseases (as subgraphs) based on a knowledge graph of symptoms (as nodes) [1]. In some cases, not all symptoms of the disease may appear or be seen, and a diagnosis is made based solely on the observed ones. The observation order of symptoms would not follow a fixed order but depend on the specific situation of the patient.

Realistic training and evaluation protocols. For training and evaluation, we create partial subgraphs by selecting nodes $V^{\text{obs}}$ from the nodes $V^{\text{sub}}$ of the full subgraph. We fix validation and test node sets with a constant size for each subgraph. This is more realistic than selecting nodes in proportion to the subgraph size (i.e., $|V^{\text{sub}}|$) in that we cannot know the exact size at evaluation. We create a new partial subgraph of fixed size for training at every iteration. As we note above, there may be a specific observation ordering of the nodes. It is natural to take these into account when constructing the partial subgraphs. Thus, we select early observed nodes if the observation is ordered, otherwise we sample the nodes randomly.

3 MODELS

We introduce the Partial Subgraph InfoMax (PSI) framework and its models. We first describe encoder-readout pipelines for learning subgraphs. Given a subgraph $(V^*, A^*)$ and features $X^{\text{glob}}$, the GNN encoder $\mathcal{E}$ outputs the node representations $H^* = [\tilde{h}_1, ..., \tilde{h}_|V^*|] \in \mathbb{R}^{|V^*| \times |F^*|}$, and the readout $\mathcal{R}$ generates the summary $\bar{s} \in \mathbb{R}^F$. Finally, the prediction function $\mathcal{F}$ computes the logit $\hat{y} \in \mathbb{R}^C$. The superscript * denotes the graph type as in §2 such as ‘sub’ for the full subgraph, ‘obs’ for the partial subgraph. This is summarized as:

$$ (V^*, A^*, X^{\text{glob}}) \xrightarrow{\mathcal{E}} H^* \xrightarrow{\mathcal{R}} \bar{s} \xrightarrow{\mathcal{F}} \hat{y}. \quad (1) $$

3.1 Partial Subgraph InfoMax framework

The encoder-readout is insufficient for partial subgraph learning since only information from observed nodes is considered for prediction. Thus, we leverage a structural hierarchy from nodes to partial and full subgraphs using mutual information (MI) maximization. The PSI framework encodes the information of the full subgraph into the partial subgraph representation by maximizing MI between the partial subgraph summary $\bar{s}^{\text{obs}}$ and representations from the full subgraph, nodes $H^{\text{sub}}$ or summary $\bar{s}^{\text{sub}}$.

Among several MI estimators modeled with neural networks [3], GAN-like divergence (GD) [26] and InfoNCE [27] estimators are widely used in InfoMax models for graphs. To maximize GAN-like divergence estimator between nodes $H^{\text{sub}}$ and the subgraph $X^{\text{obs}}$, we minimize the following loss between samples from the joint distribution $P$ and the product of marginal distributions $P \times \tilde{P}$:

$$ L_{\text{GD}} = -\mathbb{E}_P \left[ \log \sigma \left( D(\tilde{h}_s, \bar{s}^{\text{obs}}) \right) \right] - \mathbb{E}_{\tilde{P}} \left[ \log \left( 1 - \sigma \left( D(\tilde{h}_s, \bar{s}^{\text{obs}}) \right) \right) \right], \quad (2) $$

where $s$ is an input sample from an empirical distribution $P$ of the input space, $\tilde{s}$ is a negative sample from $\tilde{P}$, $\bar{s}^{\text{obs}}$ is the partial subgraph summary of $s$, $\tilde{h}_s$ is the node representation in $s$. A discriminator $D : \mathbb{R}^F \times \mathbb{R}^F \rightarrow \mathbb{R}$ computes how much $\tilde{h}_s$ and $\bar{s}$ are related. We also maximize the MI bound by minimizing InfoNCE loss,

$$ L_{\text{InfoNCE}} = \mathbb{E}_P \left[ D(\tilde{h}_s, \bar{s}^{\text{obs}}) \right] - \mathbb{E}_P \left[ \log \sum_i e^{\mathcal{F}(\tilde{h}_i, \bar{s}^{\text{obs}})} \right]. \quad (3) $$

We generalize the following InfoMax models for learning partial subgraph representations: DGI [33], InfoGraph [30], MVGRL [11], and GraphCL [34] that maximize MI between local and global structures in the graph. Since they are designed for node or graph predictions, we incorporate them into PSI by considering each partial subgraph as an independent graph. Then, we jointly minimize the InfoMax loss and the cross-entropy loss $L_{\text{cross}}$ on the logit $\hat{y}$ and label $y$. See Table 1 for the architectures of PS-prefixed PSI models.
We propose a set of negative samples within the partial subgraph. For positive samples, we compute the second PSI model, which uses the summary as input to distinguish positive and negative samples. Then, we compute the partial subgraph’s MI estimator. The score between two subgraphs can be defined as follows: \[ S_{\text{sub}} = \text{MI}(V_{\text{sub}}, \langle V_{\text{glob}} \rangle) \]

We use pre-trained embeddings from Alsentzer et al. [1]. We use readout to mean pooling after a two-layer MLP for all models except for k-hop PSI in the two-stage model, where we use the soft-attention pooling [19] after a two-layer Transformer [31]. We add the positional encoding before Transformer for ordered graphs.

### 4 EXPERIMENTS

#### 4.1 Datasets

Datasets. We experiment with three real-world datasets. FNTN (Fake News in Twitter Network; ordered) [15, 20–23] is a Twitter network (G) with news propagation trees (S), contents (\( g \)), and genuineness (y). The fake news early detection task is classifying the genuineness of news by initial nodes. EM-User (Users in EndoMondo; unordered) [1, 25] is a fitness network of workouts (G), users as subgraphs (S), and their gender (y), where the task is to profile a user’s gender with only a few logs. The global graph G of HPO-Metab (Metabolic disease in Human Phenotype Ontology; unordered) [1, 10, 18, 24] is a knowledge graph of symptoms. Each subgraph S is a collection of symptoms associated with a metabolic disease, and the label (y) is the disease type. The task is to classify the disease type, assuming only some of the symptoms are observed. Detailed statistics are reported in Table 2. We randomly split the train/val/test set of FNTN with a ratio of 70/15/15 and use public splits [1] for EM-User (70/15/15) and HPO-Metab (80/10/10).

#### 4.2 Training and evaluation settings

For both training and evaluation, we set the number of observed nodes \( |V_{\text{obs}}| \) to 4 for HPO-Metab, (the average number of nodes < 16), and 8 for FNTN and EM-User, (the average size of subgraphs > 16). Further, to see the performance change with \( |V_{\text{obs}}| \), we conduct experiments where \( |V_{\text{obs}}| \) is 8, 16, 32, and 64 for FNTN and EM-User. We also experiment with an oracle setting where all subgraphs are fully observed.

#### 4.3 Model and training details

For the encoder E, we use the two-layer GraphSAGE [9] with skip connections [12]. As an input of E, node features \( \chi_{\text{glob}} \in \mathbb{R}^{d_{\text{emb}} \times |V|} \) are trainable parameters with \( F_{\text{emb}} \) of 32 (FNTN) and 64 (others). For HPO-Metab and EM-User, we use pre-trained embeddings from Alsentzer et al. [1]. We use readout \( R \) of mean pooling after a two-layer MLP for all models except for k-hop PSI in the two-stage model, where we use the soft-attention pooling [19] after a two-layer Transformer [31]. We add the positional encoding before Transformer for ordered FNTN. For the discriminator D, we use a linear scoring [27, 33] for the GD estimator, and cosine similarity with a temperature [34] for the InfoNCE estimator. For the prediction function F, we use a single-layer neural network. If there is a subgraph-level feature \( g \in \mathbb{R}^F \), we first transform it to the vector of the same length as \( z \), concatenate it with \( z \), and feed them to the prediction layer. All models use \( F = 64 \) features, the RelU activation, dropout of 0.2 [29], and the Adam optimizer [16] with a learning rate of \( 10^{-3} \). We sample nodes in a one-hop neighborhood in the k-hop PSI (i.e., \( k = 1 \)). They are implemented with PyTorch ecosystems [7, 8, 28, 35].

#### 4.4 Two-stage Models

We use k-hop PSI + PS-DGI and + PS-InfoGraph only for two-stage models since using PS-MVGRL or PS-GraphCL as a second model is practically difficult. For MVGRL, using non-shared encoders and PPR augmentation requires significant memory and computations, and for GraphCL, a large batch size is needed.

#### 4.5 Baselines

Baselines. All baselines share the following encoder-readout architecture: \( H = E(B(X|V_{\text{obs}}), A_{\text{emb}}), z = R^B(H), \tilde{g} = F(z, [g]) \), where \( E^B \) are two-layer MLP, GCN [17], GraphSAGE [9], GAT [32], and SubGNN [1] with skip connections. We set \( R^B \) the two-layer MLP after mean pooling for SubGNN and mean pooling after two-layer MLP for others. We report the performance of SubGNN on EM-User and HPO-Metab only since SubGNN requires large memory of \( O(|V_{\text{glob}}|^2) \) (~TB for FNTN) in computing shortest paths.

### 5 RESULTS AND DISCUSSIONS

Performance by models and datasets. Table 3 summarizes the mean accuracies over five runs of various models. We confirm that PSI models outperform all comparison models for all three datasets except for PS-DGI and k-hop PSI. PS-DGI performs worse than the best baseline in EM-User and HPO-Metab. Among PSI models, PS-InfoGraph and PS-GraphCL consistently outperform baselines across datasets. PS-MVGRL shows the best performance among single PSI models on FNTN, but it does not fit in single GPU (VRAM of 11G) on EM-User. However, the performance differences among

| Table 2: Statistics of real-world datasets. |
|-------------------------------------------|
|                                           |
| # Global nodes | 362,232 | 57,333 | 14,587 |
| # Global edges | 22,918,295 | 4,573,417 | 3,238,174 |
| Density of G | 0.0002 | 0.0028 | 0.0304 |
| # Subgraphs | 1107 | 319 | 2397 |
| # Nodes per subgraph | 408.6 ± 386.7 | 155.4 ± 100.4 | 14.4 ± 6.2 |
| # Edges per subgraph | 412.9 ± 391.3 | 534.9 ± 643.5 | 181.3 ± 181.8 |
| Density of S | 0.004 ± 0.003 | 0.016 ± 0.005 | 0.758 ± 0.149 |
| # Classes | 4 | 2 | 6 |
Table 3: Mean and standard deviation of accuracy of five runs. The first column is the number of observed nodes $|\mathcal{V}^{\text{obs}}|$ (§4), and the setting of each dataset is indicated with †, ‡. The PSI model that outperforms the best baseline (except for the oracle) is indicated by color, and statistical significance by unpaired t-test by asterisk ($**p<.001$, *p < .1$).

| $|\mathcal{V}^{\text{obs}}|$ | Model     | FNTN$^\dagger$ | EM-User$^\dagger$ | HPO-Metab$^\dagger$ |
|----------------|-----------|-----------------|-------------------|-------------------|
| 100%           | GraphSAGE | 86.3$\pm$0.7   | 82.1$\pm$1.2     | 47.7$\pm$3.3     |
| 84, 87         | MLP       | 82.5$\pm$2.6   | 71.9$\pm$4.6     | 45.3$\pm$4.4     |
|                 | GCN       | 84.6$\pm$2.0   | 72.8$\pm$3.8     | 42.9$\pm$1.8     |
|                 | GraphSAGE | 84.9$\pm$1.3   | 68.1$\pm$2.6     | 44.1$\pm$1.3     |
|                 | GAT       | 85.3$\pm$0.8   | 71.5$\pm$5.7     | 43.1$\pm$2.3     |
|                 | SubGNN    | N/A             | 61.3$\pm$5.6     | 37.1$\pm$1.5     |
|                 | PS-DGI    | 87.5$\pm$2.2   | 72.3$\pm$4.2     | 44.0$\pm$1.8     |
|                 | PS-InfoGraph | 87.3$\pm$2.0   | 75.7$\pm$3.9     | 47.1$\pm$2.1     |
|                 | PS-MVGRL  | 88.6$\pm$0.9   | OOM               | 45.4$\pm$2.4     |
|                 | PS-GraphCL | 88.1$\pm$1.3   | 75.3$\pm$2.4     | 47.2$\pm$3.5     |
|                 | k-hop PSI | 87.8$\pm$2.2   | 73.5$\pm$2.4     | 42.4$\pm$2.6     |
|                 | + PS-DGI  | 88.0$\pm$0.7   | 75.7$\pm$4.4     | 45.6$\pm$1.0     |
|                 | k-hop PSI | 86.0$\pm$1.8   | 77.0$\pm$5.2     | 44.6$\pm$1.6     |
|                 | + PS-InfoGraph | 87.3$\pm$2.0   | 75.7$\pm$3.9     | 47.1$\pm$2.1     |

The PSI models except for k-hop PSI are not statistically significant ($p$-value > .1 for all datasets in one-way ANOVA).

In FNTN and EM-User, k-hop PSI is on par with other PSI models, but in HPO-Metab, k-hop PSI significantly underperforms. This behavior is caused by differences in the density of the global graph. As in Table 2, HPO-Metab has a higher density (0.03) than FNTN (2 × 10−4) and EM-User (2.8 × 10−3). When k-hop subgraph sampling is used, more neighbor nodes are included for denser graphs. Since most of the sampled neighbors are not in the subgraph, it is difficult to distinguish which of many neighbors belong to the subgraph by the discriminator $D$ in k-hop PSI. It degrades the performance of discriminator $D$ and the classification performance eventually.

Two-stage model’s performance is mainly driven by the k-hop PSI’s performance. In HPO-Metab where k-hop PSI does not perform well, the performance is lower than single PSIs; otherwise, it outperforms single models. The noise from high density is still relevant in two-stage models. In all datasets, a two-stage model results in better performance than a single k-hop PSI, and the combination with PS-InfoGraph performed better than with PS-DGI.

Finally, we discuss the results of baselines. First, SubGNN underperforms simple models. SubGNN uses message-passing between subgraphs, and thus partial observation degrades its performance. Second, there is no significance difference in MLP, GCN, GraphSAGE, and GAT ($p$-value > .1 in one-way ANOVA).

Performance by the number of observed nodes. In Figure 2a, we show the mean accuracy of k-hop PSI + PS-InfoGraph (5 runs) by the number of observed nodes in training and test. 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 [4]. Note that adding observed nodes is equivalent to adding k-hop neighbors to be discriminated by $D$ in k-hop PSI. That is, the impact of performance degradation from neighborhood noise is more significant than information gain from additional nodes in FNTN.

**Generalization across sizes of test subgraphs.** Figure 2b shows how k-hop PSI + PS-InfoGraph generalizes across sizes of test subgraphs (mean performance over 5 runs). 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 trials do not converge on EM-User.

**Sensitivity to $\lambda$.** In Figure 3, we plot the test accuracy on FNTN and EM-User against $\lambda^{k}$-hop and $\lambda^{2nd}$. 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. We also proposed a novel framework, Partial Subgraph Infomax (PSI), which maximizes the mutual information between the partial subgraph’s summary and representations of substructures like nodes or full subgraphs. Using training and evaluation protocols designed to simulate real-world use cases, PSI models outperform baselines in three datasets. One limitation is that k-hop PSI 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. We leave how to effectively and efficiently choose nodes as future work.

**ACKNOWLEDGMENTS**

This research was supported by the ERC Program through the NRF funded by the Korean Government MSIT (NRF-2018R1A5A1059921).
REFERENCES

[1] Emily Alsentzer, Samuel G Finlayson, Michelle M Li, and Marinka Zitnik. 2020. Subgraph Neural Networks. NeurIPS (2020).

[2] Peter W Battaglia, Jessica B Hanneck, Victor Bapst, Alvarez Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. 2018. Relational inductive biases, deep learning, and graph networks. arXiv preprint arXiv:1806.01261 (2018).

[3] Mohamed Ihssen Belghazi, Aristide Baratin, Sui Rajeswaran, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. 2018. Mutual information neural estimation. In ICML. PMLR, 531–540.

[4] Tian Bian, Xi Xiao, Tingyang Xu, Peilin Zhao, Wenhing Huang, Yu Rong, and Junzhou Huang. 2020. Rumor detection on social media with bi-directional graph convolutional networks. In AAAI. Vol. 34. 549–556.

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

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

[7] William Falcon and The PyTorch Lightning team. 2019. PyTorch Lightning. https://doi.org/10.5281/zenodo.3828935

[8] Matthias Fey and Jan E. Lenssen. 2019. Fast Graph Representation Learning with PyTorch Geometric. In ICLR Workshop on Representation Learning on Graphs and Manifolds.

[9] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In NeurIPS. 1025–1035.

[10] Taila Hartley, Fabrice Lareau, Kristin D Kernohan, Heather E Howley, David R Adams, and Kym M Boycott. 2020. New diagnostic approaches for undiagnosed rare genetic diseases. Annual review of genomics and human genetics 21 (2020), 351–372.

[11] Kaveh Hassani and Amir Hosein Khasahmad. 2020. Contrastive multi-view representation learning on graphs. In ICML. PMLR, 4116–4126.

[12] Kaining He, Xiangyu Zhang, Shaqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition. 770–778.

[13] R Devon Hjelm, Alex Fedorov, Samuel Lavioie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. 2019. Learning deep representations by mutual information estimation and maximization. In ICLR.

[14] Weihua Hu, Matthias Fey, Marinka Zitnik, Tuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. arXiv preprint arXiv:2005.06687 (2020).

[15] Joyooy Kim, Dongkwan Kim, and Alice Oh. 2019. Homogeneity-based trans- mission process to model true and false news in social networks. In WSDM.

[16] Diderik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014).

[17] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In ICLR.

[18] Sebastian Köhler, Leigh Carmody, Nicole Vasilevsky, Julia O B Jacobsen, Daniel Danis, Jean-Philippe Gourdin, Michael Gargano, Nomi L Harris, Nicolas Mamentzoglou, Julie A McMurty, et al. 2019. Expansion of the Human Phenotype Ontology (HPO) knowledge base and resources. Nucleic acids research 47, D1 (2019), D1018–D1027.

[19] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. 2015. Gated graph sequence neural networks. arXiv preprint arXiv:1511.05493 (2015).

[20] Xiaomo Liu, Arminneh Nourbaksh, Quanlin Li, Rui Fang, and Sameena Shah. 2015. Real-time rumor debunking on twitter. In CIKM. 1867–1870.

[21] Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting rumors from microblogs with recurrent neural networks (2016). In IJCAI 3818–3824.

[22] Jing Ma, Wei Gao, and Kam-Fai Wong. 2017. Detect Rumors in Microblog Posts Using Propagation Structure via Kernel Learning. In ACL. 708–717.

[23] Jing Ma, Wei Gao, and Kam-Fai Wong. 2018. Rumor Detection on Twitter with Tree-structured Recursive Neural Networks. In ACL. 1980–1989.

[24] Dylan Mordaunt, David Cox, and Maria Fuller. 2020. Metabolomics to improve rare genetic diseases. Nucleic acids research 47, D1 (2019), D549–D556.

[25] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. 2016. f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization. In NeurIPS. Vol. 29.

[26] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018).

[27] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In NeurIPS. 8026–8037.

[28] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research 15, 39 (2014), 1929–1958.

[29] Fan-Yun Sun, Jordan Hoffman, Vikas Verma, and Jian Tang. 2020. InfoGraph: Unsupervised and Semi-supervised Graph-Level Representation Learning via Mutual Information Maximization. In ICLR.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. NeurIPS 30 (2017), 5998–6008.

[31] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph Attention Networks. In ICLR.

[32] Petar Veličković, William Fedus, William L. Hamilton, Pietro Lio, Yoshua Bengio, and R Devon Hjelm. 2019. Deep Graph Infomax. In ICLR.

[33] Yunqing You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. NeurIPS (2020).

[34] Yanqiao You, Zichen Xu, Qiang Liu, and Shu Wu. 2021. An Empirical Study of Graph Contrastive Learning. arXiv.org (Sept. 2021). arXiv:2109.01116v1 [cs.LG]