# A Theory Details

From the $\mathcal{L}_{\mathrm{EGI}}$ objective, we have assumed $g_i \overset{i.i.d.}{\sim} \mu$, $x_i \overset{i.i.d.}{\sim} \nu$, and $(g_i, x_i) \overset{i.i.d.}{\sim} p$, for $(g_i, x_i) \in \mathcal{G} \times \mathcal{X}$. Then for a sample $\{(g_i, x_i)\}_i$, we have access to the empirical distributions of the three. In the procedure of evaluating the objective, we sample uniformly.

Note that, in Eq. 2 of the main paper, we used a $d$ dimensional hidden state $h_p$ to denote an edge's source node representation and $x_q$ as destination node features from the structure of the ego-graph and the associated source node feature with GNN. In our proof, we denote $v_{p,q}$ as the $q$-th node in the $p$-th layer of the ego-graph and let $h_{p,q} = h_p$ and $x_{p,q} = x_q$. For simplicity, in i-th layer, we denote $f(x^i) = h^i_{p,q} \| x^i_{p,q}$, where $[\cdot\|\cdot]$ is the concatenation operation.

Finally, as we are considering GNN with $k$ layers, its computation only depends on the k-hop ego-graphs of $G$, which is an important consideration when unfolding the embedding of GNN at a centre node with Lamma A.1.

**Lemma A.1.** *For any $A \in \mathbb{R}^{m \times n}$, where $m \geq n$, and $A$ is a submatrix of $B \in \mathbb{R}^{m' \times n}$, where $m < m'$, we have*

$$\|A\|_2 \leq \|B\|_2.$$

*Proof.* Note that, $AA^T$ is a principle matrix of $BB^T$, *i.e.*, $AA^T$ is obtained by removing the same set of rows and columns from $BB^T$. Then, by Eigenvalue Interlacing Theorem [24] and the fact that $A^TA$ and $AA^T$ have the same set of non-zero singular values, the matrix operator norm satisfies $\|A\|_2 = \sqrt{\lambda_{\max}(A^TA)} = \sqrt{\lambda_{\max}(AA^T)} \leq \sqrt{\lambda_{\max}(BB^T)} = \|B\|_2$. □

## A.1 Center-node view of GCN

Recall that $V_p(g_i)$ denotes the set of nodes in the $p$th hop of k-hop ego-graph $g_i$, and $x^i_{p,q}$ denotes the feature for $q$th node in $p$th hop of $g_i$, for any $p = 0, \ldots, k;\ q = 1, \ldots, |V_p(g_i)|$. Similarly, $V(g_i)$ denotes the entire set of nodes in $g_i$. In each ego-graph sample $\{g_i, x_i\}$, the layer-wise propagation rules for the center node embedding in encoder $\Psi$ and discriminator $\mathcal{D}$ can be written into the form of GCN as followed

$$Z^{(l)} = \mathrm{ReLU}(D^{-\frac{1}{2}}(I + A)D^{-\frac{1}{2}}Z^{(l-1)}\theta^{(l)})$$

where $A$ is adjacency matrix of $G$. $I$ adds the self-loop and $D_{ii} = \sum_j A_{ij}$ is the degree matrix.

We focus on the center node's embedding obtained from a $k$-layer GCN with 1-hop polynomial filter $\phi(L) = Id - L$. Inspired by the characterization of GCN from a node-wise view in [55], we similarly denote the embedding of node $x_i$ $\forall i = 1, \cdots, n$ in the final layer of the GCN as

$$z_i^{(k)} = z_i = \Psi_\theta(x_i) = \sigma(\sum_{j \in \mathcal{N}(x_i)} e_{ij} {z_j^{(k-1)}}^T \theta^{(k)}) \in \mathbb{R}^d,$$

where $e_{ij} = [\phi(L)]_{ij} \in \mathbb{R}$ the weighted link between node $i$ and $j$; and $\theta^{(k)} \in \mathbb{R}^{d \times d}$ is the weight for the $k$th layer sharing across nodes. Then $\theta = \{\theta^{(\ell)}\}_{\ell=1}^k$. We may denote $z_i^{(\ell)} \in \mathbb{R}^d$ similarly for $\ell = 1, \cdots, k-1$, and $z_i^0 = x_i \in \mathbb{R}^d$ as the node feature of center node $x_i$. With the assumption of GCN in the statement, it is clear that only the k-hop ego-graph $g_i$ centered at $x_i$ is needed to compute $z_i^{(k)}$ for any $i = 1, \cdots, n$ instead of the whole of $G$. Precisely, $p$-hop of subgraph corresponds to the $\ell = (k - p)$th layer in the model.

With such observation in mind, let us denote the matrix of node embeddings of $g_i$ at the $\ell$th layer as $[z_i^{(\ell)}] \in \mathbb{R}^{|V(g_i)| \times d}$ for $\ell = 1, \cdots, k$; and let $[z_i^{(0)}] \equiv [x_i] \in (\mathbb{R}^d)^{|V(g_i)|}$ denote the matrix of node features in the $k$-hop ego-graph $g_i$. In addition, denote $[z_i^{(\ell)}]_p$ as the principle submatrix, which includes embeddings for nodes in the 0 to $p$th hop of $g_i$, $0 \leq p \leq k$.

We denote $L_{g_i}$ as the out-degree normalised graph Laplacian of $g_i$. Here, the out-degree is defined with respect to the direction from leaves to centre node in $g_i$. Similarly, denote $\tilde{L}_{g_i}$ as the in-degree normalised graph Laplacian of $g_i$, where the direction is from centre to leaves.

WLOG, we write the $\ell$th layer embedding in matrix notation of the following form

$$[z_i^{(\ell)}]_{k-\ell+1} = \sigma([\phi(L_{g_i})]_{k-\ell+1}[z_i^{(\ell-1)}]_{k-\ell+1}\theta^{(\ell)}),$$

where the GCN only updates the embedding of nodes in the 0 to $(k-\ell)$th hop. We also implicitly assume the embedding of nodes in $(k-\ell+1)$ to $k$th hop are unchanged through the update, due to the directed nature of $g_i$. Hence, we obtain $z_i \equiv [z_i^{(k)}]_0$ from the following

$$[z_i^{(k)}]_1 = \sigma([\phi(L_{g_i})]_1[z_i^{(k-1)}]_1\theta^{(k)}).$$

Similarly, we are able to write down the form of discriminator using matrix representation for GCN. The edge information at $\ell$th time point for nodes in $V(g_i)$ can be described as follows

$$[h_i^{(\ell)}] = ReLU(\phi(\tilde{L}_{g_i})[h_i^{(\ell-1)}]\tilde{\theta}^{(\ell)}),$$

### A.2 Proof for Theorem 4.1

We restate Theorem 4.1 from the main paper as below.

**Theorem A.2.** *Let $G_a = \{(g_i, x_i)\}_{i=1}^n$ and $G_b = \{(g_{i'}, x_{i'})\}_{i'=1}^m$ be two graphs and node features are structure-respecting with $x_i = f(L_{g_i}), x_{i'} = f(L_{g_{i'}})$ for some function $f : \mathbb{R}^{|V(g_i)|\times|V(g_i)|} \rightarrow \mathbb{R}^d$. Consider GCN $\Psi_\theta$ with $k$ layers and a 1-hop polynomial filter $\phi$,the empirical performance difference of $\Psi_\theta$ with $\mathcal{L}_{\text{EGI}}$ satisfies*

$$|\mathcal{L}_{\text{EGI}}(G_a) - \mathcal{L}_{\text{EGI}}(G_b)| \leq \mathcal{O}\left(\frac{1}{nm}\sum_{i=1}^{n}\sum_{i'=1}^{m}[M + C\lambda_{\max}(L_{g_i} - L_{g_{i'}}) + \tilde{C}\lambda_{\max}(\tilde{L}_{g_i} - \tilde{L}_{g_{i'}}))]\right), \tag{7}$$

*where $M$ is dependant on $\Psi$, $\mathcal{D}$, node features, and the largest eigenvalue of $L_{g_i}$ and $\tilde{L}_{g_i}$. $C$ is a constant dependant on the encoder, while $\tilde{C}$ is a constant dependant on the decoder. With a slight abuse of notation, we denote $\lambda_{\max}(A) := \lambda_{\max}(A^TA)^{1/2}$. Note that, in the main paper, we have $C := M + C\lambda_{\max}(L_{g_i} - L_{g_{i'}})$, and $\Delta_{\mathcal{D}}(G_a, G_b) := \tilde{C}\lambda_{\max}(\tilde{L}_{g_i} - \tilde{L}_{g_{i'}})$.*

*Proof.* Now,

$$\begin{aligned}
&|\mathcal{L}_{\text{EGI}}(G) - \mathcal{L}_{\text{EGI}}(G')| \\
=&\left|\frac{1}{n^2}\sum_{i,j=1}^{n}(\mathcal{D}(g_i, z_j)) - \frac{1}{n}\sum_{i=1}^{n}(-(-\mathcal{D}(g_i, z_i)) - (\frac{1}{m^2}\sum_{i',j'=1}^{m}(\mathcal{D}(g_{i'}, z_{j'})) - \frac{1}{m}\sum_{i'=1}^{m}(-(-\mathcal{D}(g_{i'}, z_{i'}))))\right| \\
\leq&\frac{1}{n^2m^2}\sum_{i,j=1}^{n}\sum_{i',j'=1}^{m}|\mathcal{D}(g_i, z_j) - \mathcal{D}(g_{i'}, z_{j'})| + \frac{1}{nm}\sum_{i=1}^{n}\sum_{i'=1}^{m}|\mathcal{D}(g_i, z_i) - \mathcal{D}(g_{i'}, z_{i'})| \\
=&\frac{1}{n^2m^2}\sum_{i,j=1}^{n}\sum_{i',j'=1}^{m}A + \frac{1}{nm}\sum_{i=1}^{n}\sum_{i'=1}^{m}B.
\end{aligned}$$

We make the following assumptions in the proof,

1. Assume the size of the neighborhood for each node is bounded by $0 < r < \infty$, then the maximum number of node for $p$-th layer subgraph is bounded by $r^p$. WLOG, let $1 \leq |V_p(g_i)| \leq |V_p(g_{i'})| \leq r^p$;
2. Assume $h^i_{p,q}\|x^i_{p,q} = 0$ if $|V_p(g_i)| < q$, i.e. assume non-informative edge information and node features for non-existed nodes in the smaller neighborhood with no links;

From Assumption 2, we add isolated nodes to the smaller neighborhood $V_p(g_i)$ such that the neighborhood size at each hop match. It can be found in our code to compute EGI gap as pad_nbhd. For the following proof, we WLOG assume $|V_p(g_i)| = |V_p(g_{i'})|\ \forall p$.

First we consider $B$. Recall that, $V_p(g_i)$ is the set of nodes in layer $p$ of $g_i$,

$$\mathcal{D}(g_i, z_i) = \sum_{p=1}^{k}\sum_{q=1}^{|V_p(g_i)|}\log(\sigma_{sig}\left(U^T\tau\left(W^T[f(x^i)\|z_i]\right)\right)),$$

where $\sigma_{sig}(t) = \frac{1}{1+e^{-t}}$ is the sigmoid function, $\tau$ is some $\gamma_\tau$-Lipschitz activation function and $[\cdot \| \cdot]$ denotes the concatenation of two vectors. Then we obtain

$$U^T \tau \left( W^T [f(x^i) \| z_i] \right) = U^T \tau \left( W_1^T f(x^i) + W_2^T z_i \right).$$

Since $\log(\sigma_{sig}(t)) = -\log(1 + e^{-t})$, which is 1-Lipschitz, it gives

$$\begin{aligned}
B &\leq \sum_{p}^{k} \Big| \sum_{q}^{|V_p(g_{i'})|} \sigma_s(U^T \tau \left( W_1^T f(x^i) + W_2^T z_i \right)) - \sigma_s(U^T \tau \left( W_1^T f(x^{i'}) + W_2^T z_{i'} \right)) \Big| \\
&\leq \gamma_\tau \|U\|_2 \sum_{p=1}^{k} \sum_{q=1}^{|V_p(g_{i'})|} (\|W_1^T f(x^i) - W_1^T f(x^{i'})\|_2 + \|W_2^T z_i - W_2^T z_{i'}\|_2) \\
&\leq \gamma_\tau \|U\|_2 s_w \left( \sum_{p=1}^{k} \sum_{q=1}^{|V_p(g_{i'})|} \left[ \|h_{p,q}^i - h_{p,q}^{i'}\|_2 + \|x_{p,q}^i - x_{p,q}^{i'}\|_2 \right] + \sum_{p=1}^{k} \sum_{q=1}^{|V_p(g_{i'})|} \|z_i - z_{i'}\|_2 \right) \\
&\leq C_1 \left( \sum_{p=1}^{k} \sum_{q=1}^{|V_p(g_{i'})|} \left[ \|h_{p,q}^i - h_{p,q}^{i'}\|_2 + \|x_{p,q}^i - x_{p,q}^{i'}\|_2 \right] / \sum_{p=1}^{k} r^p + \|z_i - z_{i'}\|_2 \right) \\
&= C_1 \left( I_1 + I_2 \right)
\end{aligned} \tag{8}$$

We provide the derivation for the unfolding of $\ell$th layer GCN with the centre-node view in Lemma A.3. This will be used in the derivation of $I_1$ and $I_2$.

**Lemma A.3.** *For any $\ell = 1, \cdots, k$, we have an upper bound for the hidden representation difference between $g_i$ and $g_i'$,*

$$\begin{aligned}
\|[z_i^{(\ell)}]_{k-\ell} - [z_{i'}^{(\ell)}]_{k-\ell}\|_2 \leq{}& (\gamma_\sigma c_\theta)^\ell \|\phi(L_{g_i})\|_2^\ell \|[x_i] - [x_{i'}]\|_2 \\
&+ \frac{(\gamma_\sigma c_\theta)^\ell \|\phi(L_{g_i})\|_2^\ell + 1}{\gamma_\sigma c_\theta \|\phi(L_{g_i})\|_2 - 1} \gamma_\sigma c_\theta c_z \|\phi(L_{g_i}) - \phi(L_{g_{i'}})\|_2.
\end{aligned} \tag{9}$$

*Specifically, for $\ell = k$, we obtain the expansion for center node embedding $\|[z_i^{(k)}]_0 - [z_{i'}^{(k)}]_0\| \equiv \|z_i - z_{i'}\|$.*

*Proof.* By Lemma A.1, for any $\ell = 1, \cdots, k$, the following holds

$$\|[z_i^{(\ell)}]_{k-\ell} - [z_{i'}^{(\ell)}]_{k-\ell}\|_2 \leq \|[z_i^{(\ell)}]_{k-\ell+1} - [z_{i'}^{(\ell)}]_{k-\ell+1}\|_2.$$

Assume $\max_\ell \|[z_i^{(\ell)}]\|_2 \leq c_z < \infty \; \forall i$, and $\max_\ell \|\theta^{(\ell)}\|_2 \leq c_\theta < \infty$, where $c_\theta = \vee_\ell s_{\theta^{(\ell)}}$ the largest singular value.

Then, for $\ell = 1, \cdots, k-1$, we have

$$\begin{aligned}
&\|[z_{i'}^{(\ell)}]_{k-\ell} - [z_{i'}^{(\ell)}]_{k-\ell}\|_2 \\
\leq{}& \|[\sigma([\phi(L_{g_i})]_{k-\ell+1}[z_i^{(\ell-1)}]_{k-\ell+1}\theta^{(\ell)}) - \sigma([\phi(L_{g_{i'}})]_{k-\ell+1}[z_{i'}^{(\ell-1)}]_{k-\ell+1}\theta^{(\ell)})]_{k-\ell})\|_2 \\
\leq{}& \gamma_\sigma \|[\phi(L_{g_i})]_{k-\ell+1}[z_i^{(\ell-1)}]_{k-\ell+1} - [\phi(L_{g_{i'}})]_{k-\ell+1}[z_{i'}^{(\ell-1)}]_{k-\ell+1}\|_2 \|\theta^{(k)}\|_2 \\
\leq{}& \gamma_\sigma c_\theta \|[\phi(L_{g_i})]_{k-\ell+1}\|_2 \|[z_i^{(\ell-1)}]_{k-\ell+1} - [z_{i'}^{(\ell-1)}]_{k-\ell+1}\|_2 + \gamma_\sigma c_\theta \|[z_{i'}^{(\ell-1)}]_{k-\ell+1}\|_2 \|[\phi(L_{g_i})]_{k-\ell+1} - [\phi(L_{g_{i'}})]_{k-\ell+1}\|_2 \\
\leq{}& \gamma_\sigma c_\theta \|\phi(L_{g_i})\|_2 \|[z_i^{(\ell-1)}]_{k-\ell+1} - [z_{i'}^{(\ell-1)}]_{k-\ell+1}\|_2 + \gamma_\sigma c_\theta c_z \|\phi(L_{g_i}) - \phi(L_{g_{i'}})\|_2.
\end{aligned} \tag{10}$$

since $[\phi(L_{g_i})]_{k-\ell+1}$ is the principle submatrix of $\phi(L_{g_i})$. Then we equivalently write the above equation as $E_\ell \leq b E_{\ell-1} + a$, which gives

$$E_\ell \leq b^\ell E_1 + \frac{b^\ell + 1}{b - 1} a.$$

With $[x_i] = [z_i^{(0)}]_k$, we see the following is only dependant on the structure of $g_i$ and $g_{i'}$,

$$\|[z_{i'}^{(\ell)}]_{k-\ell} - [z_{i'}^{(\ell)}]_{k-\ell}\|_2 \leq (\gamma_\sigma c_\theta)^\ell \|\phi(L_{g_i})\|_2^\ell \|[x_i] - [x_{i'}]\|_2 + \frac{(\gamma_\sigma c_\theta)^\ell \|\phi(L_{g_i})\|_2^\ell + 1}{\gamma_\sigma c_\theta \|\phi(L_{g_i})\|_2 - 1} \gamma_\sigma c_\theta c_z \|\phi(L_{g_i}) - \phi(L_{g_{i'}})\|_2.$$

$\square$

Since the the graph Laplacians are normalised, we have $\|\phi(L_{g_i})\|_2 \leq c_L < \infty \;\forall i$. In addition, let

$$\|x_{p,q}^i - x_{p,q}^{i'}\|_2 \leq \sup_i \sup_{p,q} \|x_{p,q}^i - x_{p,q}^{i'}\|_2 = \sup_i \|f(L_{g_i}) - f(L_{g_{i'}})\|_2 := \delta_x.$$

Hence, $\|[x_i] - [x_{i'}]\|_2 \leq \delta_x (\sum_{p=1}^k r^p)^{1/2} := c_x$. From Lemma A.3, it is clear that we obtain the following at the final layer

$$\begin{aligned} I_2 = \|z_i - z_{i'}\|_2 &\leq (\gamma_\sigma c_\theta c_L)^k c_x + \frac{(\gamma_\sigma c_\theta c_L)^k + 1}{\gamma_\sigma c_\theta c_L - 1} \gamma_\sigma c_\theta c_z \|\phi(L_{g_i}) - \phi(L_{g_{i'}})\|_2 \\ &\leq C(M c_x + \|L_{g_i} - L_{g_{i'}}\|_2) \\ &= C(M c_x + \lambda_{\max}(L_{g_i} - L_{g_{i'}})^{1/2}). \end{aligned} \tag{11}$$

since $\phi$ is a linear function for $L$. Indeed, this can be generalised to polynomial function $\phi$ of higher powers.

Now, consider the following term that is related with discriminator $\mathcal{D}$,

$$I_1 = \sum_{p=1}^k \sum_{q=1}^{|V_p(g_{i'})|} \Big[ \|h_{p,q}^i - h_{p,q}^{i'}\|_2 + \|x_{p,q}^i - x_{p,q}^{i'}\|_2 \Big] / \sum_{p=1}^k r^p$$

Firstly, we denote $\tilde{L}_{p,q}$ as the in-degree graph Laplacian derived with the subgraph $g_q$ of $g_i$ centred at $q \in V_p(g_i)$. Different from the encoder, we utilize every node's hidden embedding in the computation. Specifically, $g_q$ is obtained by retrieving links in $g_i$ that connects to the $q$th node in the $p$th layer. This is a principle submatrix of the in-degree graph Laplacian $\tilde{L}_{g_i}$ of $g_i$.

Just as defined in §A.1, we denote $[h_q^{(p)}]_\ell$ as the $p$th layer GCN embedding for nodes in hop 0 to hop $\ell \in [0, p]$ of $g_q$. Note that in this case, $[h_q^{(p)}]_0 = h_q^{(p)}$, which is one row of $[h_i^{(p)}]$, corresponding to the $q$-th node in the neighborhood. So we may write the first term in $I_1$ as

$$\sum_{p=1}^k \sum_{q=1}^{|V_p(g_{i'})|} \|h_q^{(p)} - h_{q'}^{(p)}\|$$

where $h_{q'}^{(p)} := h_{p,q}^{i'}$ for short. In this way, we regard each node $q \in V_p(g_i)$ as the centre node, which allows us to unfold the convolution similarly as expanding the $I_2$ term. Now, for any $q \in V_k(g_i)$, i.e. when $p = k$, we apply Lemma A.3 similarly as for $\|z_i - z_{i'}\|_2$. Then,

$$\begin{aligned} \|h_q^{(k)} - h_{q'}^{(k)}\| &\leq (\gamma_\sigma c_{\tilde{\theta}} c_{\tilde{L}})^k c_x + \frac{(\gamma_\sigma c_{\tilde{\theta}} c_{\tilde{L}})^k + 1}{\gamma_\sigma c_{\tilde{\theta}} c_{\tilde{L}} - 1} \gamma_\sigma c_{\tilde{\theta}} c_h \|\phi(\tilde{L}_{k,q}) - \phi(\tilde{L}_{k,q'})\|_2 \\ &\leq \tilde{C}_k(\tilde{M}_k c_x + \|\phi(\tilde{L}_{g_i}) - \phi(\tilde{L}_{g_{i'}})\|_2) \end{aligned}$$

where $\tilde{L}_{p,q}$ is the principle submatrix of $\tilde{L}_{g_i}$ and Lemma A.1 can be applied iin the last inequality. In addition, $\tilde{C}_k$ and $\tilde{M}_k$ are taken to be the maximum over any $q \in V_k(g_i)$. In general, for $q \in V_p(g_i)$, $0 < p < k$, we have

$$\|h_q^{(p)} - h_{q'}^{(p)}\|_2 \leq \tilde{C}_p(\tilde{M}_p c_x + \|\phi(\tilde{L}_{g_i}) - \phi(\tilde{L}_{g_{i'}})\|_2)$$

Take a common upper bound for $\tilde{C}_p, \tilde{M}_p$ over $0 < p \leq k$, we obtain

$$\begin{aligned} \sum_{p=1}^k \sum_{q=1}^{|V_p(g_{i'})|} \|h_q^{(p)} - h_{q'}^{(p)}\| / \sum_{p=1}^k r^p &\leq \tilde{C}(\tilde{M} c_x + \|\tilde{L}_{g_i} - \tilde{L}_{g_{i'}}\|_2) \\ &= \tilde{C}(\tilde{M} c_x + \lambda_{\max}(\tilde{L}_{g_i} - \tilde{L}_{g_{i'}})^{1/2}) \end{aligned}$$

In addition, for the other half of $I_1$, we have

$$\sum_{p=1}^{k} \sum_{q=1}^{|V_p(g_{i'})|} \|x_{p,q}^{i} - x_{p,q}^{i'}\|_2 / \sum_{p=1}^{k} r^p \leq \sup_i \sup_{p,q} \|x_{p,q}^{i} - x_{p,q}^{i'}\|_2 = \delta_x = c_x / (\sum_{p=1}^{k} r^p)^{1/2}$$

We can write $\mathcal{B}$ in terms of weights $C$ and $\tilde{C}$, which is dependant on the activation function $\sigma$, $k$ and $\sup_i \lambda_{\max}(L_{g_i})$. Hence,

$$\begin{aligned} B &\leq (CM + \tilde{C}\tilde{M} + 1/(\sum_{p=1}^{k} r^p))c_x + C\lambda_{\max}(L_{g_i} - L_{g_{i'}}) + \tilde{C}\lambda_{\max}(\tilde{L}_{g_i} - \tilde{L}_{g_{i'}}) \\ &= M'c_x + C\lambda_{\max}(L_{g_i} - L_{g_{i'}}) + \tilde{C}\lambda_{\max}(\tilde{L}_{g_i} - \tilde{L}_{g_{i'}}) \end{aligned}$$

Note that the derived $I_1$ for $B$ is the same for $A$, since the node features, edge information and embedded features are bounded by separate terms in Eq. 8. The only difference is given by $I_2$, where a different set of graph Laplacians $L_{g_j}$, $L_{g_{j'}}$ and node features $(x_j)$ are used. Therefore,

$$A \leq M'c_x + C\lambda_{\max}(L_{g_j} - L_{g_{j'}}) + \tilde{C}\lambda_{\max}(\tilde{L}_{g_i} - \tilde{L}_{g_{i'}})$$

Hence the result. $\square$

Note that, our view of structural information is closely related to graph kernels [4] and graph perturbation [55]. Specifically, our Definition on k-hop ego-graph is motivated by the concept of k-layer expansion sub-graph in [4]. However, [4] used the Jensen-Shannon divergence between pairwise representations of sub-graphs to define a depth-based sub-graph kernel, while we depict $G$ as samples of its ego-graphs. In this sense, our view is related to the setup in [55], which derived a uniform algorithmic stability bound of a 1-layer GNN under 1-hop structure perturbation of $G$.

In the setting of domain adaptation, [7] draws a connection between the difference in the distributions of source and target domains and the model transferability, and learns a transferable model by minimizing such distribution differences. This coincides with our approach of connecting the structure difference of two graphs in terms of k-hop subgraph distributions and the transferability of GNNs in the above theory.

# B Model Details

Following the same notations used in the main paper, EGI consists of a GNN encoder $\Psi$ and a GNN discriminator $\mathcal{D}$. In general, the GNN encoder $\Psi$ and discriminator $\mathcal{D}$ can be any existing GNN models. For each ego-graph and its node features $\{g_i, x_i\}$, the GNN encoder returns node embedding $z_i$ for the center node $v_i$. As mentioned in Eq. 2 in the main paper, the GNN discriminator $\mathcal{D}$ makes edge-level predictions as follows,

$$\mathcal{D}(e_{\tilde{v}v} | h_{p,q}^{\tilde{q}}, x_{p,q}^{i}, z_i) = \sigma\left(U^T \cdot \tau\left(W^T [h_{p,q}^{\tilde{q}} || x_{p,q}^{i} || z_i]\right)\right), \tag{12}$$

where $e_{\tilde{v}v} \in E(g_i)$ and $h_{p,q}^{\tilde{q}} \in \mathbb{R}^d$ (simplified as $h_p$ in the main paper, same for $x_{p,q}^{i} = x_q$) is the representation for edge $e_{\tilde{v}v}$ between node $v_{p-1,\tilde{q}}$ in hop $p-1$ and $v_{p,q}$ in hop $p$. The prediction relies on the combination of center node embedding $z_i$, destination node feature $x_{p,q}^{i}$ and source node representation $h_{p,q}^{\tilde{q}}$. And now we describe how we calculate the source node representation in $\mathcal{D}$.

To obtain the source node representation representations $h$, the GNN in discriminator $\mathcal{D}$ operates on a reversed ego-graph $\tilde{g}_i$ while encoder $\Psi$ performs forward propagation on $g_i$. The discriminator GNN starts from the center node $v_i$ and compute the hidden representation $m_{p-1,\tilde{q}}$ for node $v_{p-1,q}$ at each hop. We denote the source node at $p-1$ hop as $\tilde{q} \in \tilde{Q}_{p,q}, \tilde{Q}_{p,q} = \{\tilde{q} : v_{p-1,\tilde{q}} \in V_{p-1}(g_i), e_{(p-1,\tilde{q})(p,q)} \in E(g_i)\}$. Although $h_{p,q}$ is calculated as node embedding, in reversed ego graph $\tilde{g}_i$, node only has one incoming edge. Thus, we can also interpret $h_{p,q}^{\tilde{q}}$ as the edge embedding as it combines source node's hidden representation $m_{p-1,\tilde{q}}$ and destination node features $x_{p,q}$ as follows,

$$h_{p,q}^{\tilde{q}} = \text{ReLU}\left(W_p^T \left(m_{p-1,\tilde{q}} + x_{p,q}^{i}\right)\right), \; m_{p-1,\tilde{q}} = \frac{1}{|\tilde{Q}_{p-1,\tilde{q}}|} \sum_{q' \in \tilde{Q}_{p-1\tilde{q}}} h_{p-1,\tilde{q}}^{q'} \tag{13}$$

**Algorithm 1:** Pseudo code for training EGI

```
The GNN encoder $\Psi$ and the GNN discriminator $\mathcal{D}$, k-hop ego graph and features $\{g_i, x_i\}$;
/* EGI-training starts */
while $\mathcal{L}_{\text{EGI}}$ not converges do
    Sample M ego-graphs $\{(g_1, x_1), ..., (g_M, x_M)\}$ from empirical distribution $\mathbb{P}$ without
      replacement, and obtained their positive and negative node embeddings $z_i, z_i'$ through $\Psi$
          $z_i = \Psi(g_i, x_i), z_i' = \Psi(g_i', x_i'),$
    /* Initialize positive and negative expectation in Eq. 1 in the main paper*/
    $E_{pos} = 0, E_{neg} = 0$
    for $p$ = 1 to $k$ do
        /* Compute JSD on edges at each hop*/
        for $e_{(p-1,\tilde{q})(p,q)} \in E(g_i)$ do
            generate source node embedding $h^{\tilde{q}}_{p,q}$ in Eq. 13 ;
            $E_{\text{pos}} = E_{\text{pos}} + \sigma\left(U^T \cdot \tau\left(W^T[h^{\tilde{q}}_{p,q}||x^i_{p,q}||z_i]\right)\right)$
            $E_{\text{neg}} = E_{\text{neg}} + \sigma\left(U^T \cdot \tau\left(W^T[h^{\tilde{q}}_{p,q}||x^i_{p,q}||z_i']\right)\right)$
        end
    end
    /* Compute batch loss*/
    $\mathcal{L}_{\text{EGI}} = E_{\text{neg}} - E_{\text{pos}}$
    /* Update $\Psi$, $\mathcal{D}$ */
    $\theta_\Psi \xleftarrow{+} -\nabla_\Psi \mathcal{L}_{\text{EGI}}$, $\theta_{\mathcal{D}} \xleftarrow{+} -\nabla_{\mathcal{D}} \mathcal{L}_{\text{EGI}}$
end
```

When $p = 1$, every edge origins from the center node $v_i$ and $m_{0,q'}$ is the center node feature $x_{v_i}$. Note that we the elaborated aggregation rule is equivalent as layer-wise propagation rules (different in-degree matrix for each $h_{p,q}$) of EGI earlier in §A.1.

In every batch, we sample a set of ego-graphs and their node features $\{g_i, x_i\}$. During the forward pass of encoder $\Psi$, it aggregates from neighbor nodes to the center node $v_i$. Then, the discriminator calculates the edge embedding in Eq. 13 from center node $v_i$ to its neighbors and make edge-level predictions–*fake* or *true*. Besides training framework Figure 2 in the main paper, the algorithm EGI is depicted in Algorithm 1.

We implement our method and all of the baselines using the same encoders $\Psi$: 2-layer GIN [60] for synthetic and role identification experiments, 2-layer GraphSAGE [15] for the relation prediction experiments. We set hidden dimension as 32 for both synthetic and role identification experiments, For relation prediction fine-tuning task, we set hidden dimension as 256. We train EGI in a mini-batch fashion since all the information for encoder and discriminators are within the k-hop ego-graph $g_i$ and its features $x_i$. Further, we conduct neighborhood sampling and set maximum neighbors as 10 to speed up the parrallel training. The space and time complexity of EGI is $O(BN^K)$, where $B$ is the batch size, $N$ is the number of the neighbors and k is the number of hops of ego-graphs. Notice that both the encoder $\Psi$ and discriminator $\mathcal{D}$ propagate message on the k-hop ego-graphs, so the extra computation cost of $\mathcal{D}$ compared with a common GNN module is a constant multiplier over the original one. The scalability of EGI on million scale YAGO network is reported in section C.3.

### B.1 Transfer Learning Settings

The goal of transfer learning is to train a model on a dataset or task, and use it on another. In our graph learning setting, we focus on training the model on one graph and using it on another. In particular, we focus our study on the setting of *unsupervised-transfering*, where the model learned on the source graph is directly applied on the target graph without *fine-tuning*. We study this setting because it allows us to directly measure the transferability of GNNs, which is not affected by the fine-tuning process on the target graph. In other words, the fine-tuning process introduces significant uncertainty to the analysis, because there is no guarantee on how much the fine-tuned GNN is different from the pre-trained one. Depending on specific tasks and labels distributions on the two graphs, the fine-tuned

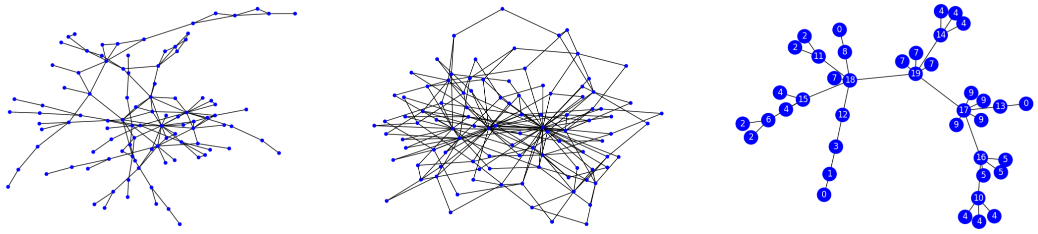
GNN might be quite similar to the pre-trained one, or it can be significantly different. It is then very hard to analyze how much the pre-trained GNN itself is able to help. Another reason is about efficiency. The fine-tuning of GNNs requires the same environment set-up and computation resource as training GNNs from scratch, although it may take less training time eventually if pre-training is effective. It is intriguing if this whole process can be eliminated when we guarantee the performance with unsupervised-transfering.

In our experiments, we also study the setting of transfer learning with fine-tuning, particularly on the real-world large-scale YAGO graphs. Since we aim to study the general transferability of GNNs not bounded to specific tasks, we always pre-train GNNs with the unsupervised pre-training objective on source graphs. Then we enable two types of fine-tuning. The first one is *post-fine-tuning* ($\mathcal{L} = \mathcal{L}_s$), where the pre-trained GNNs are fine-tuned with the supervised task specific objective $\mathcal{L}_s$ on the target graphs. The second on is *joint-fine-tuning* ($\mathcal{L} = \mathcal{L}_s + \mathcal{L}_u$), where pre-training is the same, but fine-tuning is done *w.r.t.* both the pre-training objective $\mathcal{L}_u$ and task specific objective $\mathcal{L}_s$ on target graphs in a semi-supervised learning fashion. The unsupervised pre-training objective $\mathcal{L}_u$ of EGI is Algorithm 1, while those of the compared algorithms are as defined in their papers. The supervised fine-tuning objective $\mathcal{L}_s$ is the same as in the DistMult paper [61] for all algorithms.

# C Additional Experiment Details

## C.1 Synthetic Experiments

**Data.** As mentioned in the main paper, we use two traditional graph generation models for synthetic data generation: (1) barabasi-albert graph [5] and (2) forest-fire graph [32]. We generate 40 graphs each with 100 nodes with each model. We control the parameters of two models to generate two graphs with different ego-graph distributions. Specifically, we set the number of attached edges as 2 for barabasi-albert model and set $p_{\text{forward}} = 0.4$, $p_{\text{backward}} = 0.3$ for forest-fire model. In Figure 4a and 4b, we show example graphs from two families in our datasets. They have the same size but different appearance which leads to our study on the transferability gap $\Delta_{\mathcal{D}}$(F, F) and $\Delta_{\mathcal{D}}$(B, F) in Table 1 in the main paper. The accuracy of this task defined as the percentage of nearest neighbors for target node in the embedding space $z = \Psi(\cdot)$ that are structure-equivalent, *i.e.* #correct k-nn neighbors / #ground truth equivalent nodes.



**(a)** Forest-fire graph example **(b)** Barabasi-albert graph example **(c)** structural label example

**Figure 4:** Visualizations of the graphs and labels we use in the synthetic experiments.

**Results.** The structural equivalence label is obtained by a 2-hop WL-test [58] on the ego-graphs. If two nodes have the same 2-hop ego-graphs, they will be assigned the same label. In the example of Figure 4c, the nodes labeled with same number (*e.g.* 2, 4) have the isomorphic 2-hop ego-graphs. Note that this task is exactly solvable when node features and GNN architectures are powerful enough like GIN [60]. In order to show the performance difference among different methods, we set the length of one-hot node degree encoding to 3 (all nodes with degrees higher than 3 have the same encoding). Here, we present the performance comparison with different length of degree encodings (d) in Table 4. When the capacity of initial node features is high (d=10), the transfer learning gap diminishes between different methods and different graphs because the structural equivalence problem can be exactly solved by neighborhood aggregations. However, when the information in initial node features is limited, the advantage of EGI in learning and transfering the graph structural information is obvious. In Table 5, we also show the performance of different transferable and non-transferable features discussed after Definition 4.3 in the main paper, *i.e.* node embedding [42] and random feature vectors. The observation is similar with Table 1 in the main paper: the transferable feature can reflect the performance gap between similar and dissimilar graphs while non-transferable features can not.

In both Table 4 and 8 here as well as Table 1 in the main paper, we report the structural difference among graphs in the two sets ($\bar{d}$) calculated *w.r.t.* the term $\Delta_{\mathcal{D}}(G_a, G_b)$ on the RHS of Theorem 4.1 in the main paper. This indicates that the Forest fire graphs are structurally similar to the other Forest fire graphs, while less similar to the Barabasi graphs, as can be verified from Figure 4a and 4b. Our bound in Theorem 4.1 then tells us that the GNNs (in particular, EGI) should be more transferable in the F-F case than B-F. This is verified in Table 4 and 5 when using the transferable node features of degree encoding with limited dimension (d=3) as well as DeepWalk embedding, as EGI pre-trained on Forest fire graphs performs significantly better on Forest fire graphs than on Barabasi graphs (with +0.094 and +0.057 differences, respectively).

**Table 4:** Synthetic experiments of identifying structural-equivalent nodes with different degree encoding dimensions.

| Method | **#dim degree encoding d = 3** | | | **# dim degree encoding d = 10** | | | **structural difference** | |
|---|---|---|---|---|---|---|---|---|
| | F-F | B-F | $\delta$(acc.) | F-F | B-F | $\delta$(acc.) | $\Delta_D$(F,F) | $\Delta_D$(B,F) |
| GCN (untrained) | 0.478 | 0.478 | / | 0.940 | 0.940 | / | 0.752 | 0.883 |
| GIN (untrained) | 0.572 | 0.572 | / | 0.940 | 0.940 | / | | |
| VGAE (GIN) | 0.498 | 0.432 | +0.066 | 0.939 | 0.937 | 0.002 | | |
| DGI (GIN) | 0.578 | 0.591 | -0.013 | 0.939 | 0.941 | -0.002 | | |
| EGI (GIN) | **0.710** | 0.616 | +0.094 | 0.942 | 0.942 | 0 | | |

**Table 5:** Synthetic experiments of identifying structural-equivalent nodes with different transferable and non-transferable features.

| Method | **DeepWalk embedding** | | | **random vectors** | | | **structural difference** | |
|---|---|---|---|---|---|---|---|---|
| | F-F | B-F | $\delta$(acc.) | F-F | B-F | $\delta$(acc.) | $\Delta_D$(F,F) | $\Delta_D$(B,F) |
| GCN (untrained) | 0.658 | 0.658 | / | 0.246 | 0.246 | / | 0.752 | 0.883 |
| GIN (untrained) | 0.663 | 0.663 | / | 0.520 | 0.520 | / | | |
| GVAE (GIN) | 0.713 | 0.659 | +0.054 | 0.266 | 0.264 | 0.002 | | |
| DGI (GIN) | 0.640 | 0.613 | +0.027 | 0.512 | 0.576 | -0.064 | | |
| EGI (GIN) | **0.772** | 0.715 | +0.057 | 0.507 | 0.485 | +0.022 | | |

## C.2 Real-world Role Identification Experiments

**Data.** We report the number of nodes, edges and classes for both airport and gene dataset. The numbers for the Gene dataset are the aggregations of the total 52 gene networks in the dataset. For the three airport networks, Figure 5 shows the power-law degree distribution on log-log scale. The class labels are between 0 to 3 reflecting the level of the airport activities [45]. For the Gene dataset, we matched the gene names in the TCGA dataset [68] to the list of transcription factors on wikipedia[5]. 75% of the genes are marked as 1 (transcription factors) and some gene graphs have extremely imbalanced class distributions. So we conduct experiments on the relatively balanced gene graphs

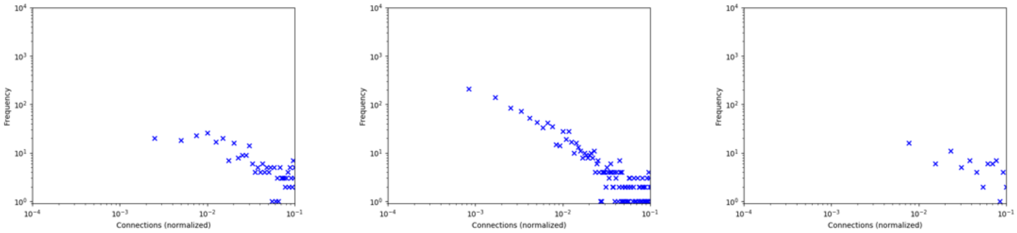
**Table 6:** Overall Dataset Statistics

| Dataset | # Nodes | # Edges | # Classes |
|---|---|---|---|
| Europe | 399 | 5,995 | 4 |
| USA | 1,190 | 13,599 | 4 |
| Brazil | 131 | 1,074 | 4 |
| Gene | 9,228 | 57,029 | 2 |

---

[5] `https://en.wikipedia.org/wiki/Transcription_factor`

of brain cancers (Figure 2 in the main paper). Both datasets do not have organic node attributes. The role-based node labels are highly relevant to their local graph structures, but are not trivially computable such as from node degrees.



**(a)** Europe airport log-log plot **(b)** USA airport log-log plot **(c)** Brazil airport log-log plot

**Figure 5:** Visualizations of power-law degree distribution on three airport dataset.

**Results.** As we can observe from Figure 5, the three airport graphs have quite different sizes and structures (*e.g.*, regarding edge density and connectivity pattern). Thus, the absolute classification accuracy in both Table 2 in the main paper and Table 8 here varies across different graphs. However, as we mention in the main paper, the structural difference we compute based on Eq. 5 in Theorem 3.1 is close among the Europe-USA and Europe-Brazil graph pairs (0.869 and 0.851), which leads to close transferability of EGI from Europe to USA and Brazil. This indicates the effectiveness of our view over essential structural information. We also provide detailed standard deviations of Table 2 (main paper) when using node degree as features.

**Table 7:** Results of role identification with direct-transfering on the Airport dataset (Table 2, main paper). The performance reported (%) are the average over 100 runs. We set all node features same as non-transferable features.

| Method | Europe (source) | | USA (target) | | Brazil (target) | |
|---|---|---|---|---|---|---|
| | node degree | same feat. | node degree | same feat. | node degree | same feat. |
| features | 52.81 | 20.59 | 55.67 | 20.22 | 67.11 | 19.63 |
| GIN (untrained) | 55.75 | 53.88 | 61.56 | 58.32 | 70.04 | 70.37 |
| GVAE | 53.90 | 21.12 | 55.51 | 22.39 | 66.33 | 17.70 |
| DGI | 57.75 | 22.13 | 54.90 | 21.76 | 67.93 | 18.78 |
| MaskGNN | 56.37 | 55.53 | 60.82 | 54.64 | 66.71 | 74.54 |
| ContextPredGNN | 52.69 | 49.95 | 50.38 | 54.75 | 62.11 | 70.66 |
| Structural Pre-train | 56.00 | 53.83 | 62.17 | 57.49 | 68.78 | 72.41 |
| MVC | 53.16 | 51.69 | 59.66 | 50.42 | 66.07 | 61.55 |
| GMI | 58.12 | 46.25 | 59.28 | 47.64 | 73.07 | 62.96 |
| EGI (GIN) | **59.15**** | 54.98 | **64.55**** | 57.40 | **73.15** | 70.00 |

Besides that, the results present in Table 8 are the accuracy of GNNs directly trained and evaluated on each network without transfering. Therefore, only the Europe column has the same results as in Table 2 in the main paper, while the USA and Brazil columns can be regarded as providing an upper-bound performance of GNN transfered from other graphs. As we can see, EGI gives the closest results from Table 2 (main paper) to Table 8 here, demonstrating the its plausible transferability. The scores are so close, showing a possibility to skip fine-tuning when the source and target graphs are similar enough. Also note that, although the variances are pretty large (which is also observed in other works like [45] since the networks are small), our t-tests have shown the improvements of EGI to be significant.

### C.3 Real-world large-scale Relation Prediction Experiments

**Data.** As shown in Table 9, the source graph we use to pre-train GNNs is the full graph cleaned from the YAGO dump [49], where we assume the relations among entities are unknown. The target

**Table 8:** Role identification that identifies structurally similar nodes on real-world networks. The performance reported are the average and standard deviation for 10 runs. Our classification accuracy on three datasets all passed the t-test (p<0.01) with the second best result in the table.

| Method | **Airport [45]** | | |
|---|---|---|---|
| | Europe | USA | Brazil |
| node degree | 52.81% ± 5.81% | 55.67% ± 3.63% | 67.11% ± 7.58% |
| GCN (random-init) | 52.96% ± 4.51% | 56.18% ± 3.82% | 55.93% ± 1.38% |
| GIN (random-init) | 55.75% ± 5.84% | 62.77% ± 2.35% | 69.26% ± 9.08% |
| GVAE (GIN) | 53.90% ± 4.65% | 58.99% ± 2.44% | 55.56% ± 6.83% |
| DGI (GIN) | 57.75% ± 4.47% | 62.44% ± 4.46% | 68.15% ± 6.24% |
| Mask-GIN | 56.37% ± 5.07% | 63.78% ± 2.79% | 61.85% ± 10.74% |
| ContextPred-GIN | 52.69% ± 6.12% | 56.22% ± 4.05% | 58.52% ± 10.18% |
| Structural Pre-train | 56.00% ± 4.58% | 62.29% ± 3.51% | 71.48% ± 9.38 % |
| MVC | 53.16% ± 4.07% | 62.81 % ± 3.12% | 67.78 % ± 4.79% |
| GMI | 58.12 % ± 5.28% | 63.36 % ± 2.92% | 73.70% ± 4.21% |
| EGI (GIN) | **59.15% ± 4.44%** | **65.88% ± 3.65%** | **74.07% ± 5.49%** |

**Table 9:** dataset statistics and running time of EGI

| Dataset | # Nodes | # Edges | # Relations | # Train/Test | Training time per epoch |
|---|---|---|---|---|---|
| YAGO-Source | 579,721 | 2,191,464 | / | / | 338 seconds |
| YAGO-Target | 115,186 | 409,952 | 24 | 480/409,472 | 134 seconds |

graph we use is a subgraph uniformed sampled from the same YAGO dump (we sample the nodes and then include all edges among the sampled nodes). The similar ratio between number of nodes and edges can be observed in Table 9. On the target graph, we also have the access to 24 different relations [48] such as *isAdvisedBy*, *isMarriedTo* and so on. Such relation labels are still relevant to the graph structures, but the relevance is lower compared with the structural role labels. We use the 256-dim degree encoding as node features for pre-training on the source graph, then we use the 128-dim positional embedding generated by LINE [51] for fine-tuning on the target graph, to explicitly make the features differ across source and target graphs.

**Results.** In Section B.1, we introduced two different types of fine-tuning, *i.e.*, *post-fine-tuning* and *joint-fine-tuning*. For both types of fine-tuning, we add one feature encoder $\mathcal{E}$ before feeding it into the GNNs for two purposes. First, the target graph fine-tuning feature usually has different dimensions with the pre-training features, such as the node degree encoding we use. Second, the semantics and distributions of fine-tuning features can be different from pre-training features. The feature encoder aims to bridge the gap between feature difference in practice. The supervised loss used in this experiment is the same as in DistMult [61]. In particular, the bilinear score function is calculated as $s(h, r, t) = z_h^T M_r z_t$, where $M_r$ is a diagonal matrix for each relation $r$, $z_h$ and $z_t$ the the embedding of GNN encoder $\Psi$ for head and tail entities. The experiments were run on GTX1080 with 12G memories. We report the average training time per epoch of our algorithm in pre-training and fine-tuning stage in Table 9 as well. The pre-training and fine-tuning takes about 40 epochs and 10 epochs to converge, respectively. In Table 9, we also present the per-epoch training time of EGI. EGI takes about 338 seconds per epoch for optimizing the ego-graph information maximization objective on YAGO-source. As we can see, fine-tuning also takes significant time compared to pre-training, which strengthens our arguments about avoiding or reducing fine-tuning through structural analysis. We implement all baselines within the same pipeline, and the running times are all in same scale.

### C.4 Parameter study

In this section, we provide additional parameter analysis towards proposed EGI model - choices of $k$, and efficiency study on EGI gap $\Delta_{\mathcal{D}}$ - sampling frequencies.

**Performance of different size of ego-graphs.** In our Theorem 3.1 and EGI algorithm (Eq. 1), number of hops $k$ determines the size of ego-graphs. In principle, $k$ may affect the transferability of EGI in two ways: (1) larger $k$ may make the EGI model (both center node encoder $\Psi$ and neighbor node encoder $\Phi$) more expressive (better precision) and the EGI gap $\Delta_{\mathcal{D}}$ more accurate (better

**Table 10:** Comparison of EGI with different $k$. Accuracy and EGI gap $\Delta_{\mathcal{D}}$ are reported.

| | Europe (source) | USA (target) | | Brazil (target) | |
|---|---|---|---|---|---|
| | acc. | acc. | $\Delta_{\mathcal{D}}$ | acc. | $\Delta_{\mathcal{D}}$ |
| EGI (k=1) | 58.25 | 60.08 | 0.385 | 60.74 | 0.335 |
| EGI (k=2) | 59.15 | 64.55 | 0.869 | 73.15 | 0.851 |
| EGI (k=3) | 57.63 | 64.12 | 0.912 | 72.22 | 0.909 |

predictiveness); (2) However, the GNN encoders may suffer from the over-smoothing problem and the computations may suffer from more noises. Therefore, it is hard to determine the influence of $k$ without empirical analysis. As we can observe in , when $k = 1$ or $k = 3$, the classification accuracy of the source graph is worse than $k = 2$, likely because the GNN encoder is either less powerful or over-smoothed. As a result, $k = 2$ obtains the best transferability to both the USA and Brazil networks. When $k = 3$, $\Delta_{\mathcal{D}}$ likely accounts for too subtle/noisy ego-graph differences and may become less effective in predicting the transferability. Therefore, we choose $k = 2$ to conduct experiments in main paper.

**Precision of $\Delta_{\mathcal{D}}$ under different sampling frequencies.** In Table 11, we present the estimated $\Delta_{\mathcal{D}}$ versus sampling frequency for 10 runs on airport dataset. A theoretical study on its convergence could be an interesting future direction. As we can observe, large sample frequency leads to more accurate and robust estimation of $\Delta_{\mathcal{D}}$. Between Europe and USA, although 100 pairs of ego-graphs are only equivalent as 2.1% of the total pair-wise enumerations, the estimated $\Delta_{\mathcal{D}}$ is pretty close.

**Table 11:** EGI gap $\Delta_{\mathcal{D}}$ on airport dataset with different sampling frequencies.

| Sampling frequency | $\Delta_{\mathcal{D}}$(Europe, USA) | $\Delta_{\mathcal{D}}$(Europe, Brazil) |
|---|---|---|
| 100 pairs | 0.872±0.039 | 0.854±0.042 |
| 1000 pairs | 0.859±0.012 | 0.848±0.007 |
| All pairs | 0.869 ±0.000 | 0.851 ±0.000 |

## Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] Mainly see Sections 3 and 4
   (b) Did you describe the limitations of your work? [Yes] See Section 5
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 5
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Section 3.2 and Appendix §A
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix §A
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See Section 4 and supplemental material
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 4
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Due to space limit, but we conducted significant tests on all claimed improvements
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 4
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 4
   (b) Did you mention the license of the assets? [No] They are all public
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We provide our models and code in the supplemental material
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]