Unsupervised Network Embedding Beyond Homophily

Anonymous authors
Paper under double-blind review

Abstract

Network embedding (NE) approaches have emerged as a predominant technique to represent complex networks and have benefited numerous tasks. However, most NE approaches rely on a homophily assumption to learn embeddings with the guidance of supervisory signals, leaving the *unsupervised heterophilous* scenario relatively unexplored. This problem becomes especially relevant in fields where a scarcity of labels exists. Here, we formulate the unsupervised NE task as an r-ego network discrimination problem and develop the SELENE framework for learning on networks with homophily and heterophily. Specifically, we design a dual-channel feature embedding pipeline to discriminate r-ego networks using node attributes and structural information separately. We employ negative-sample-free self-supervised learning objective functions to optimise the framework to learn intrinsic node embeddings. We show that SELENE's components improve the quality of node embeddings, facilitating the discrimination of connected heterophilous nodes. Comprehensive empirical evaluations on both synthetic and real-world datasets with varying homophily ratios validate the effectiveness of SELENE in homophilous and heterophilous settings showing an up to 13.35% clustering accuracy gain.

1 Introduction

Network embedding (NE) has become a predominant approach to finding effective data representations of complex systems that take the form of networks (Cui et al., 2019). NE approaches leveraging graph neural networks (GNNs) (Defferrard et al., 2016; Kipf & Welling, 2017; Xu et al., 2019b) have proven effective in (semi)-supervised settings and achieved remarkable success in various application areas, such as social, e-commerce, biology, and traffic networks (Zhang et al., 2020). However, recent works demonstrated that classic supervised NE methods powered by GNNs, which typically follow a homophily assumption, have limited representation power on heterophilous networks (Zhu et al., 2020; Bo et al., 2021; Lim et al., 2021; Zheng et al., 2022).

Whereas similar nodes are connected in homophilous networks, the opposite holds for heterophilous networks in which connected nodes are likely from different classes (Figure 1-(a-b)). For instance, people tend to connect with people of the opposite gender in dating networks (Zhu et al., 2020) and fraudsters are more likely to connect with customers than other fraudsters in online transaction networks (Pandit et al., 2007).

Traditional GNNs typically fail in heterophilous scenarios because they obtain representations by aggregating information from neighbours, acting as a low-frequency filter which generates indistinguishable node representations on heterophilous networks (Figure 2) (Bo et al., 2021). Recently, several GNN operators have been introduced to overcome the smoothing effect of traditional GNNs on heterophilous networks (Zheng et al., 2022), however, they rely heavily on the (semi)-supervised setting. In contrast, the effectiveness of GNNs in unsupervised settings, i.e. learning effective heterophilous representations without any supervision, is relatively unexplored (Xia et al., 2021). In application areas such as biomedical problems, where scarcity of labels exists, the unsupervised setting is of high interest to generate representations for various downstream tasks (Li et al., 2021).

Current limitations. We address the task of node clustering under an unsupervised heterophilous setting (Figure 1-(b-c)) (for convenience, we will refer to unsupervised network embedding as NE in the remainder of

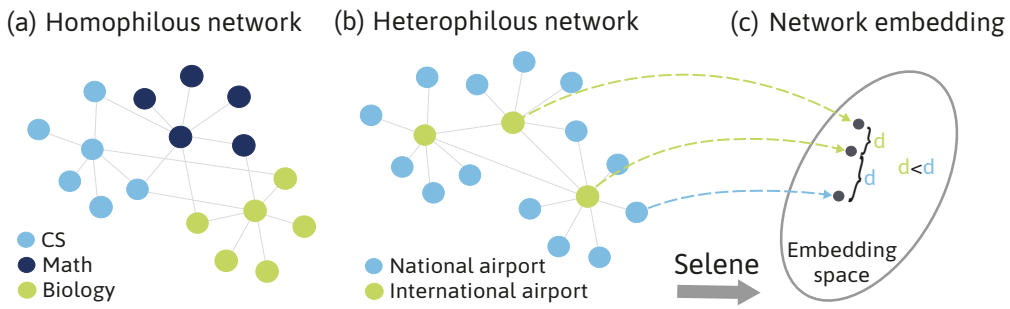

Figure 1: Example homophilous and heterophilous networks ((a): a citation network; (b): an airline transport network). (c) SELENE performs unsupervised network embedding capturing node attributes and structural information to address heterophily.

the text). In this scenario, the first unexplored question to investigate is **RQ1**: *how do existing NE methods perform on heterophilous networks without supervision?*

We conduct an empirical study on 10 synthetic networks with a variety of homophily ratios (h, Definition 3) to investigate whether h influences the node clustering performance of representative NE methods. Our experimental results, summarised in Figure 3, show that *(i)* the performance of NE methods that utilise network structure, including heterophilous GNNs designed for supervised settings, decreases significantly when $h \to 0$; *(ii)* the performance of a NE method that only relies on raw node attributes is not affected by changes of h, but it is outperformed by other NE methods when $h \to 1$. These two findings directly answer RQ1 and meanwhile raise another interesting and challenging question **RQ2:** *could we design a NE framework that adapts well to both homophily and heterophily settings under no supervision?*

Our approach. Motivated by the limitations mentioned above, we approach the NE task as an r-ego network discrimination problem and propose the SELf-supErvised Network Embedding (SELENE) framework. Out empirical study suggests that both node attributes and local structure should be leveraged to obtain node embeddings Therefore, we summarise them into an r-ego network, and use self-supervised learning (SSL) objective functions to optimise the framework to compute distinguishable node embeddings. Specifically, we propose a dual-channel embedding pipeline that encodes node attributes and network structure information in parallel (Figure 4). Next, after revisiting representative NE mechanisms, we introduce identity and network structure features to enhance the framework's ability to capture structural information and distinguish different sampled r-ego networks. Lastly, since network sampling strategies often implicitly follow homophily assumptions, i.e., "positively sampling nearby nodes and negatively sampling the faraway nodes" (Yang et al., 2020), we employ negative-sample-free SSL objective functions, namely, reconstruction loss and Barlow-Twins loss, to optimise the framework.

Extensive evaluation. We empirically evaluate our model and competitive NE methods on both synthetic and real networks covering the full spectrum of low-to-high homophily and various topics. We observe that SELENE achieves significant performance gains in both homophily and heterophily in real networks, with an up to 13.35% clustering accuracy gain. Our detailed ablation study confirms the effectiveness of each design component. In synthetic networks ($h \in [0, \ldots, 0.9]$), we observe that SELENE shows better generalisation under both homophily and heterophily settings.

Summary of contributions. To summarise, our work makes the following contributions:

- To the best of our knowledge, this is the first work that systematically studies the influence of the homophily ratio on NE methods' performance. We reveal the limitations of NE methods on learning over heterophilous networks.
- We present the SELf-supErvised Network Embedding (SELENE) framework[1], which adapts well to both homophilous and heterophilous networks.

[1]Code and data are available at this anonymised link: `https://www.dropbox.com/sh/hvc79agc6f9orxo/AABD-H9SCMc4wjMJFIqg1-VEa?dl=0`

- Extensive experiments on 12 real-world datasets and 10 synthetic networks demonstrate SELENE's superior performance on challenging node clustering tasks. A series of ablation studies confirm the effectiveness of SELENE's components.

2 Related Work

Network embedding before GNNs. NE techniques aim at embedding the node attributes and structure of complex networks into low-dimensional node representations (Cui et al., 2019). Initially, NE was posed as the optimisation of an embedding lookup table directly encoding each node as a vector. Within this group, several methods based on skip-grams (Mikolov et al., 2013) have been proposed, such as DeepWalk (Perozzi et al., 2014), node2vec (Grover & Leskovec, 2016), struc2vec (Ribeiro et al., 2017), etc (Tang et al., 2015; Qiu et al., 2018). Despite the relative success of these NE methods, they often ignore the richness of node attributes and only focus on the network structural information, which hugely limits their performance.

Network embedding with GNNs. Recently, GNNs have shown promising results in modelling structural and relational data (Wu et al., 2021). GNN models capture the structural similarity of nodes through a recursive message-passing scheme, using neural networks to implement the message, aggregation, and update functions (Battaglia et al., 2018). The effectiveness of GNNs has been widely proven in (semi)-supervised settings and they have achieved remarkable success in various areas (Zhang et al., 2020). Several approaches such as ChebNet (Defferrard et al., 2016), GCN (Kipf & Welling, 2017), GraphSAGE (Hamilton et al., 2017), CayleyNets (Levie et al., 2017), GWNN (Xu et al., 2019a), and GIN (Xu et al., 2019b) have led to remarkable breakthroughs in numerous fields in (semi-)supervised settings. However, their effectiveness for unsupervised NE is relatively unexplored. Recently, GNN-based methods for unsupervised NE such as DGI (Velickovic et al., 2019), GMI (Peng et al., 2020), SDCN (Bo et al., 2020), and GBT (Bielak et al., 2021) have been proposed, although they were primarily designed for homophilous networks.

Heterophilous network embedding. Recent works have also focused on NE for heterophilous networks and have shown that the representation power of GNNs designed for (semi-)supervised settings is greatly limited on heterophilous networks (Zheng et al., 2022). Some efforts have been dedicated to generalising GNNs to heterophilous networks by introducing complex operations, such as coarsely aggregating higher-order interactions or combining the intermediate representations (Zhu et al., 2020; Bo et al., 2021; Lim et al., 2021). Nevertheless, these heterophilous GNNs have a heavy reliance on supervisory information and hence cannot be applied to unsupervised settings (verified in Section 4 and Section 6.3).

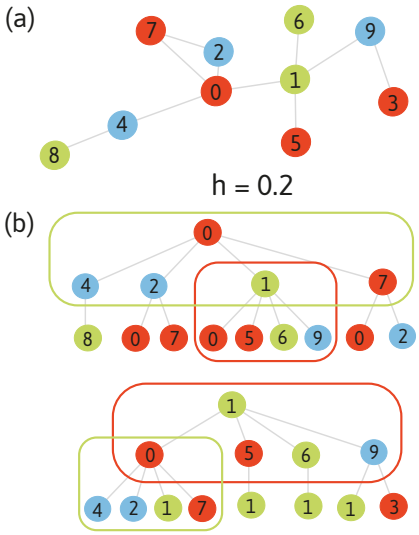

Figure 2: (a) Network with $h = 0.2$; (b) the aggregation mechanism in traditional GNNs implicitly follows a homophily assumption as a result of duplicated aggregation trees for nearby nodes - as shown for trees rooted in nodes v_0 and v_1 (duplicates marked by same-coloured square).

3 Notation and Preliminaries

An unweighted network can be formally represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathcal{V}$ is the set of nodes and $|\mathcal{V}| = n$ is the number of nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, and $\mathbf{X} \in \mathbb{R}^{n \times \pi}$ represents the π-dimensional node features. We let $\mathcal{Y} = \{y_v\}$ be a set of class labels for all $v \in \mathcal{V}$. For simplicity, we summarise $\mathcal{E}$ with an adjacency matrix $\mathbf{A} \in \{0, 1\}^{n \times n}$.

Problem setup. This paper focuses on the unsupervised node clustering task. We first learn node representations $\mathbf{Z}_v \in \mathbb{R}^d$ for all $v \in \mathcal{V}$, capturing both node attributes and local network structure. Then, the goal is to infer the unknown class labels y_v for all $v \in \mathcal{V}$ using a clustering algorithm on the learned node representation $\mathbf{Z}_v$ (in this paper, we use the well-known K-means algorithm (Hartigan & Wong, 1979)). Note that, for convenience, in the following we refer to unsupervised NE simply as NE.

Definition 1 (r-hop Neighbourhood $\mathcal{N}^r$) *We denote the r-hop neighbourhood of node v by $\mathcal{N}_v^r = \{v : d(u,v) \leq r\}$, where $d(u,v)$ is the shortest path distance between u and v. For the network shown in Figure 2-(a), $\mathcal{N}_{v_0}^1 = \{v_0, v_1, v_2, v_4, v_7\}$.*

Definition 2 (r-ego Network) *$\mathcal{G}_r(v)$) (McAuley & Leskovec, 2012; Qiu et al., 2020) Let $\mathcal{N}_v^r \subseteq \mathcal{V}$ be the r-ego neighbours of node v in $\mathcal{G}$. Their corresponding r-ego network is an induced sub-network of $\mathcal{G}$ defined as $\mathcal{G}_r(v) = \{\mathcal{N}_v^r, \mathcal{E}_v^r, \mathbf{X}_v^r\}$, where $\mathcal{E}_v^r := ((\mathcal{N}_v^r \times \mathcal{N}_v^r) \cap \mathcal{E})$.*

Definition 3 (Homophily Ratio h) *The homophily ratio h of $\mathcal{G}$ describes the relation between node labels and network structure. Recent works commonly use two measures of homophily, edge homophily (h_{edge}) (Zhu et al., 2021) and node homophily (h_{node}) (Pei et al., 2020), which can be formulated as*

$$h_{edge} = \frac{|\{(u,v) : (u,v) \in \mathcal{E} \wedge y_u = y_v\}|}{|\mathcal{E}|} \qquad h_{node} = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} \frac{|\{u : u \in \mathcal{N}_v^1 \wedge y_u = y_v\}|}{|\mathcal{N}_v^1|} \tag{1}$$

Specifically, where h_{edge} evaluates the fraction of edges between nodes with the same class labels; h_{node} evaluates the overall fraction of neighbouring nodes that have the same class labels. In this paper, we focus only on edge homophily and denote it with $h = h_{edge}$. Figure 2-(a) shows an example network with $h = 0.2$.

4 An Experimental Investigation

In this section, we empirically analyse the performance of NE methods on 10 synthetic networks with different homophily ratios (h). The main goal is to investigate (**RQ1**): *how do existing NE methods perform on heterophilous networks?* Specifically, we quantify their performance on the node clustering task on 10 synthetic networks with $h \in [0, 0.1, \ldots, 0.9]$. A detailed description of the synthetic datasets generation process can be found in Section 6.1, and we refer the reader to Section 6.2 for details on the experimental settings.

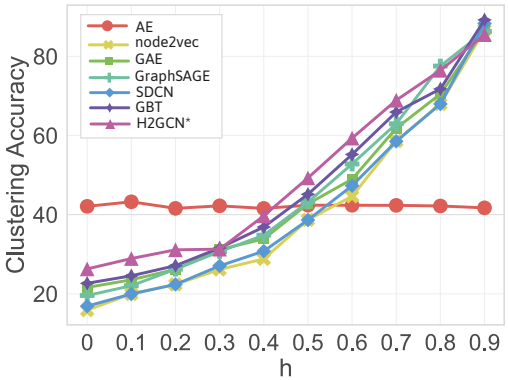

Figure 3: Node clustering accuracy of representative NE methods on synthetic networks.

Figure 3 illustrates that the clustering accuracy of representative NE methods that utilise network structure, i.e., node2vec (Grover & Leskovec, 2016), GAE (Kipf & Welling, 2016), GraphSAGE (Hamilton et al., 2017), SDCN (Bo et al., 2020), GBT (Bielak et al., 2021) and H2GCN* (Zhu et al., 2020) show outstanding performance when $h \to 1$ but with the decrease of h, their performance decreases significantly. The reason why existing NE methods using network structure fail only when $h \to 0$ is that most of them implicitly follow a homophily assumption, with specific objective function or aggregation mechanism designs. For instance, *(i)* objective functions of node2vec and GraphSAGE guide nodes at close distance to have similar representations and nodes far away to have different ones; *(ii)* the inherent aggregation mechanism of GAE, GraphSAGE, SDCN and GBT naturally assumes local smoothing (Chen et al., 2020; Min et al., 2020) (which is mainly caused by the duplicated aggregation tree for nearby nodes as shown in Figure 2-(b)), which translates into neighbouring nodes having similar representations.

On the other hand, H2GCN*, which is specifically designed for heterophilous networks, follows the same trend as the homophilous approaches due to the loss of supervisory signals in the network embedding task. The only method with stable performance across different values of h is AE (Hinton & Salakhutdinov, 2006), which is attributable to its reliance on raw node attributes only. AE exhibits an apparent advantage against all other models when $h < 0.5$. This highlights the importance of considering node attributes in the design of NE approaches for networks with heterophily.

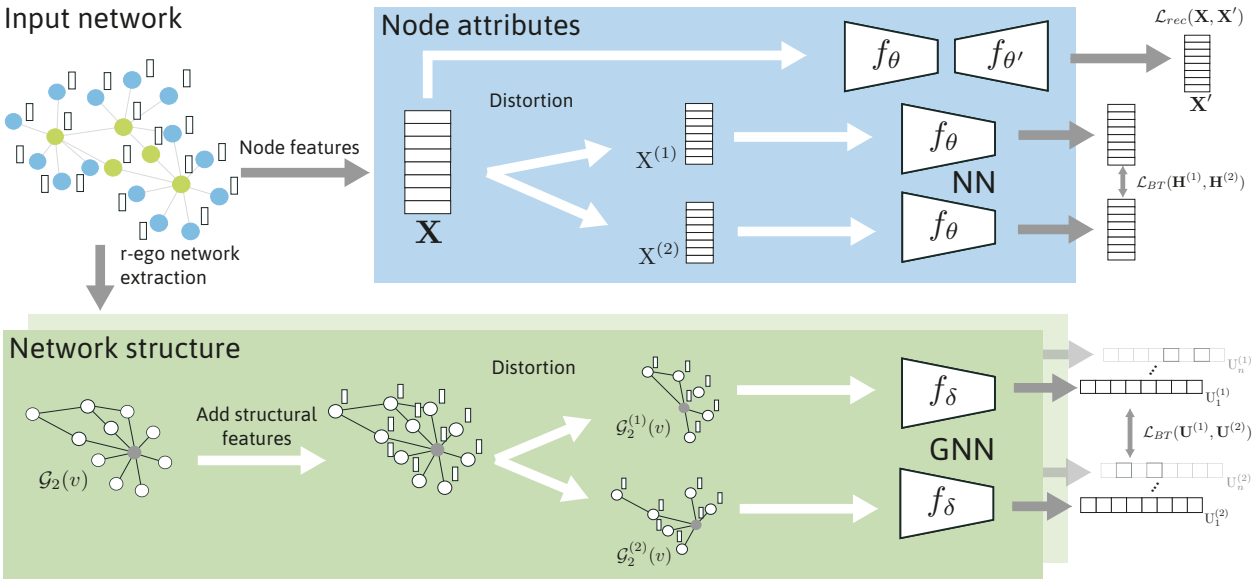

Figure 4: We propose a dual-channel feature embedding pipeline to learn node representations separately from node attributes and network structure. Node feature attribute embeddings are optimised using an autoencoder trained to reconstruct the input data $\mathcal{L}_{Rec}(\mathbf{X}, \widehat{\mathbf{X}})$, and refined through the contrastive loss $\mathcal{L}_{BT}(\mathbf{H}^{(1)}, \mathbf{H}^{(2)})$. The network structure encoder is optimised using $\mathcal{L}_{BT}(\mathbf{U}^{(1)}, \mathbf{U}^{(2)})$. The final node representation is obtained by applying COMBINE to the generated node attribute representation $\mathbf{H}$ and network structure representation $\mathbf{U}$.

5 Network Embedding via *r*-ego Network Discrimination

In this section, we formalise the main challenges of NE on heterophilous networks. To address these challenges, we present the SELf-supErvised Network Embedding (SELENE) framework. Figure 4 shows the overall view of SELENE.

Challenges. Motivated by the empirical results in Section 4, we realise that a NE method for heterophilous networks should have the ability to distinguish nodes with different attributes or structural information. Each node's local structure can be flexibly defined by its r-ego network. We further summarise each node's relevant node attribute and structural information into an r-ego network and define the NE task as an r-ego network discrimination problem. Then, we address three main research challenges to solve this problem:

RC1 How to leverage node attributes and network structure for NE?

RC2 How to break the inherent homophily assumptions of traditional NE mechanisms?

RC3 How to define an appropriate objective function to optimise the embedding learning process?

The following subsections discuss our solutions to address these three challenges and introduce the SELENE framework. In Section 6, we provide a comprehensive empirical evaluation on both synthetic and real data with varying homophily ratios to validate the effectiveness of SELENE under homophilous and heterophilous settings. We also show that all components in our design are helpful in improving the quality of node embeddings.

5.1 Dual-channel Feature Embedding (RC1)

To learn node embeddings that can discriminate r-ego networks of different nodes, we first extract the r-ego network ($\mathcal{G}_r(v)$) of each node $v \in \mathcal{G}$. For example, in Figure 4, $\mathcal{G}_2(v)$ represents a 2-ego network instance of $\mathcal{G}$. The empirical analysis of Section 4 highlighted that node attributes and structural information play a major

role in discriminating nodes over networks. Therefore, we propose a dual-channel feature embedding pipeline to learn node representation from node attributes and network structure separately, as shown in Figure 4. That said, we split r-ego network of each node $\mathcal{G}_r(v) = \{\mathcal{N}_v^r, \mathcal{E}_v^r, \mathbf{X}_v^r\}$ into ego node attribute $\mathbf{X}_v$ and network structure $\widetilde{\mathcal{G}}_r(v) = \{\mathcal{N}_v^r, \mathcal{E}_v^r\}$. Such a design brings two main benefits: *(i)* both sources of information can be well utilised without interfering with each other, and *(ii)* the inherent homophily assumptions of NE methods can be greatly alleviated, an issue we will address in the following subsection.

5.2 *r*-ego Network Feature Extraction (RC2)

Node attribute encoder module. As previously mentioned, learning effective node attribute representations is of great importance for NE. In this paper, we employ the basic Autoencoder (Hinton & Salakhutdinov, 2006) to learn representations of raw node attributes, which can be replaced by more sophisticated encoders (Masci et al., 2011; Makhzani et al., 2015; Malhotra et al., 2016) to obtain higher performance. We assume an L-layers Autoencoder (f_θ), with the formulation of the ℓ-th encoding layer being:

$$\mathbf{H}_e^{(\ell)} = \phi(\mathbf{W}_e^{(\ell)}\mathbf{H}_e^{(\ell-1)} + \mathbf{b}_e^{(\ell)}) \tag{2}$$

where ϕ is a non-linear activation function such as ReLU (Nair & Hinton, 2010) or PReLU (He et al., 2015). $\mathbf{H}_e^{(\ell-1)} \in \mathbb{R}^{n\times d_{\ell-1}}$ is the hidden node attribute representations in layer $\ell-1$, with $d_{\ell-1}$ being the dimensionality of this layer's hidden representation. $\mathbf{W}_e^{(\ell)} \in \mathbb{R}^{d_{\ell-1}\times d_\ell}$ and $\mathbf{b}_e^{(\ell)} \in \mathbb{R}^{d_\ell}$ are trainable weight matrix and bias of the ℓ-th layer in the encoder. Node representations $\mathbf{H}_v = f_\theta(\mathbf{X}_v) = \mathbf{H}_e^{(L)}$ are obtained after successive application of L encoding layers.

Identity and network structure features. Despite the significant success of GNNs in a variety of network-related tasks, their representation power in network structural representation learning is limited (Xu et al., 2019b). In order to obtain invariant node structural representations so that nodes with different ego network structures are assigned different representations, we employ the identity feature (You et al., 2021) and structural features (Li et al., 2020). In this paper, we adopt variants shortest path distance (SPD) as node structural features ($\widetilde{\mathbf{X}}_{struc}$). After, we further inject the node identity features ($\widetilde{\mathbf{X}}_{id}$) as augment features, hence we have node feature matrix $\widetilde{\mathbf{X}} = \widetilde{\mathbf{X}}_{struc} + \widetilde{\mathbf{X}}_{id}$ for each ego-network.

Network structure encoder module. Over the past few years, numerous GNNs have been proposed to learn node representations from network-structured data, including spectral GNNs (i.e., ChebNet (Defferrard et al., 2016), CayletNet (Levie et al., 2017) and GWNN (Xu et al., 2019a)) and spatial GNNs (i.e., GraphSAGE (Hamilton et al., 2017), GAT (Velickovic et al., 2018), GIN (Xu et al., 2019b)). For the sake of simplicity, we adopt a simple GNN variant, i.e., GCN (Kipf & Welling, 2017), as the building block of the network structure encoder (f_δ). The ℓ-th layer of a GCN for v's r-ego network can be formally defined as:

$$\mathbf{U}^{(\ell)} = \sigma(\widehat{\mathbf{D}}^{-\frac{1}{2}}\widehat{\mathbf{A}}\widehat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{U}^{(\ell-1)}\mathbf{W}^{(\ell)}) \tag{3}$$

with $\widehat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, where $\mathbf{I}$ is the identity matrix, and $\widehat{\mathbf{D}}$ is the diagonal node degree matrix of $\widehat{\mathbf{A}}$. $\mathbf{U}^{(\ell-1)} \in \mathbb{R}^{n\times d_{\ell-1}}$ is the hidden representation of nodes in layer $\ell-1$, with $d_{\ell-1}$ being the dimensionality of this layer's representation, and $\mathbf{U}^0 = \widetilde{\mathbf{X}}$. $\mathbf{W}^{(\ell)} \in \mathbb{R}^{d_{\ell-1}\times d_\ell}$ is a trainable parameter matrix. σ is a non-linear activation function such as ReLU or Sigmoid (Han & Moraga, 1995) function. Structural representations $\mathbf{U}_v = f_\delta(\widetilde{\mathcal{G}}_r(v)) = \mathbf{U}^{(L)}$ are obtained after successive applications of L layers.

5.3 Non-negative Self-Supervised Learning (RC3)

The objective function plays a significant role in NE tasks. Several objective functions have been proposed for NE, such as network reconstruction loss (Kipf & Welling, 2016), distribution approximating loss (Perozzi et al., 2014) and node distance approximating loss (Hamilton et al., 2017). Nevertheless, none of them is suitable for NE optimisation on heterophilous networks because of the homophily assumptions used to determine (dis)similar pairs. In heterophilous networks, node distance on network alone does not determine (dis)similarity, i.e., connected nodes are not necessarily similar, and nodes far apart are not necessarily dissimilar. This removes the need for connected nodes to be close in the embedding space and for disconnected nodes to be far apart in the embedding space.

Inspired by the latest success of non-negative self-supervised learning (SSL), we adopt the Barlow-Twins (BT) (Zbontar et al., 2021) as our overall optimisation objective. Consequently, the invariance term makes the representation invariant to the distortions applied; the redundancy reduction term lets the representation units contain non-redundant information about the target sample. More details about the BT method can be found in Appendix E.

Distortion. To apply the BT method to optimise our framework, we have to first generate distorted samples for target r-ego network instances. Due to the dual-channel feature embedding pipeline, we introduce $f_{Aug}^{\mathbf{X}}$ and $f_{Aug}^{\mathcal{G}}$ to generate distorted node attribute and network structure instances, respectively. Our data distortion method adopts a similar strategy as You et al. (2020); Bielak et al. (2021) that randomly masks node attributes and edges with probability p_x, p_e. Formally, this distortion process can be represented as:

$$\begin{aligned} f_{Aug}^{\mathbf{X}}(\mathbf{X}_v, p_x) &= (\mathbf{X}_v^{(1)}, \mathbf{X}_v^{(2)}) \\ f_{Aug}^{\mathcal{G}}(\widetilde{\mathcal{G}}_r(v), p_x, p_e) &= (\widetilde{\mathcal{G}}_r^{(1)}(v), \widetilde{\mathcal{G}}_r^{(2)}(v)) \end{aligned} \tag{4}$$

Barlow-Twins loss function. Based on the two pairs of distorted instances, two pairs of representations ($\mathbf{U}_v^{(1)}$, $\mathbf{U}_v^{(2)}$) and ($\mathbf{H}_v^{(1)}$, $\mathbf{H}_v^{(2)}$) can be computed by applying f_θ and f_δ, respectively. Then, BT method can be employed to evaluate the computed representations and guide the framework optimisation. Using representations ($\mathbf{H}^{(1)}$, $\mathbf{H}^{(2)}$) as example, the loss value is formally computed as:

$$\mathcal{L}_{BT}(\mathbf{H}^{(1)}, \mathbf{H}^{(2)}) = \sum_{i}^{|\mathcal{V}|}(1 - \mathcal{C}_{ii})^2 + \lambda \sum_{i}^{|\mathcal{V}|} \sum_{j \neq i}^{|\mathcal{V}|} \mathcal{C}_{ij}^2 \tag{5}$$

where $\mathcal{C}_{ij} = \frac{\sum_b \mathbf{H}_{b,i}^{(1)} \mathbf{H}_{b,j}^{(2)}}{\sqrt{\sum_b (\mathbf{H}_{b,j}^{(1)})^2} \sqrt{\sum_b (\mathbf{H}_{b,j}^{(2)})^2}}$. $\lambda > 0$ defines the trade-off between the invariance and redundancy reduction terms, b is the batch indexes, and i, j index the vector dimension of the input representation vectors. We adopt the default settings as Zbontar et al. (2021).

Node attribute reconstruction loss function. In addition to the general objective function, $\mathcal{L}_{BT}$, we adopt another objective function, i.e., node attribute reconstruction loss $\mathcal{L}_{Rec}$, to optimise the node attribute encoder specifically. Following the node attribute encoder (Eq.2), the decoder reconstructs input node attributes from the computed node representations $\mathbf{H}$. Typically, a decoder has the same structure as the encoder by reversing the order of layers. Its ℓ-th fully connected layer can be formally represented: $\mathbf{H}_d^{(\ell)} = \phi(\mathbf{W}_d^{(\ell)} \mathbf{H}_d^{(\ell-1)} + \mathbf{b}_d^{(\ell)})$. Reconstructed node attributes $\widehat{\mathbf{X}} = \mathbf{H}_d^{(L)}$ are obtained after successive applications of L decoding layers. We optimise the autoencoder parameters by minimising the difference between raw node attributes $\mathbf{X}$ and reconstructed node attributes $\widehat{\mathbf{X}}$ with:

$$\mathcal{L}_{Rec}(\mathbf{X}, \widehat{\mathbf{X}}) = \frac{1}{2|\mathcal{V}|} ||\mathbf{X} - \widehat{\mathbf{X}}||_F^2 \tag{6}$$

Empowered with Barlow-Twins and node attribute reconstruction loss, we can optimise the framework's encoders under heterophilous settings. The node attribute encoder is optimised with $\mathcal{L}_{BT}$ (Eq. 5) and $\mathcal{L}_{Rec}$ (Eq. 6), and the network structure encoder is optimised with $\mathcal{L}_{BT}$ (Eq. 5). The overall loss function is $\mathcal{L} = \mathcal{L}_{BT}(\mathbf{U}^{(1)}, \mathbf{U}^{(2)}) + \mathcal{L}_{BT}(\mathbf{H}^{(1)}, \mathbf{H}^{(2)}) + \mathcal{L}_{Rec}(\mathbf{X}, \widehat{\mathbf{X}})$.

Final node representations. Representations capturing node attributes ($\mathbf{H}$) and structural context ($\mathbf{U}$) are combined to obtain expressive and powerful representations as: $\mathbf{Z} = \text{COMBINE}(\mathbf{H}, \mathbf{U})$ where COMBINE($\cdot$) can be any commonly used operation in GNNs (Xu et al., 2019b), such as *mean*, *max*, *sum* and *concat*. We utilise *concat* in all our experiments, allowing for an independent integration of representations learnt by the dual-channel architecture.

5.4 Summary

Here, we summarise SELENE in Algorithm 1 to provide a general overview of our framework. Given as input a graph $\mathcal{G}$ and node attribute encoder f_θ and decoder $f_{\theta'}$, network structure encoder f_δ, node attribute

Algorithm 1: SELf-supErvised Network Embedding (SELENE) Framework

Input: Network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$;
Node attribute encoder f_θ and decoder $f_{\theta'}$;
Network structure encoder (GNN) f_δ ;
Node attribute distortion function $f^{\mathbf{x}}_{Aug}$ and network distortion function $f^{\mathcal{G}}_{Aug}$;
Output: node representations $\mathbf{Z}$

1 Sample a set of r-ego networks $\{\mathcal{G}_r(1), \mathcal{G}_r(2), \ldots, \mathcal{G}_r(n)\}$ from $\mathcal{G}$;
2 Extract network structure of each r-ego network $\mathcal{G}_r(v)$ and enhance each with identity features ($\widetilde{\mathbf{X}}_{id}$) and network structure features ($\widetilde{\mathbf{X}}_{struc}$) to obtain $\widetilde{\mathcal{G}}_r(v) = \{\mathcal{N}^r_v, \mathcal{E}^r_v, \widetilde{\mathbf{X}}\}$;
3 Generate distorted node attribute matrix instances $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$;
4 Generate two distorted instances $(\widetilde{\mathcal{G}}^{(1)}_r(v), \widetilde{\mathcal{G}}^{(2)}_r(v))$ for each r-ego network $\widetilde{\mathcal{G}}_r(v)$;
5 **repeat**
6 Initialise loss $\mathcal{L}$ as zero ;
7 **for** *each node* $v \in \mathcal{V}$ **do**
8 $\mathbf{H}_v = f_\theta(\mathbf{X}_v)$, $\widehat{\mathbf{X}}_v = f_{\theta'}(\mathbf{H}_v)$;
9 $\mathbf{H}^{(1)}_v = f_\theta(\mathbf{X}^{(1)}_v)$, $\mathbf{H}^{(2)}_v = f_\theta(\mathbf{X}^{(2)}_v)$;
10 $\mathbf{U}^{(1)}_v = f_\delta(\widetilde{\mathcal{G}}^{(1)}_r(v))$, $\mathbf{U}^{(2)}_v = f_\delta(\widetilde{\mathcal{G}}^{(2)}_r(v))$;
11 $\mathcal{L} = \mathcal{L}_{BT}(\mathbf{U}^{(1)}_v, \mathbf{U}^{(2)}_v) + \mathcal{L}_{BT}(\mathbf{H}^{(1)}_v, \mathbf{H}^{(2)}_v) + \mathcal{L}_{Rec}(\mathbf{X}_v, \widehat{\mathbf{X}}_v)$;
12 **end**
13 Update θ and δ by descending the gradients $\nabla_{\theta,\delta}\mathcal{L}$;
14 **until** *Convergence*;
15 $\mathbf{H} = f_\theta(\mathbf{X})$;
16 **for** *each node* $v \in \mathcal{V}$ **do**
17 $\mathbf{U}_v = f_\delta(\widetilde{\mathcal{G}}_r(v))$;
18 **end**
19 $\mathbf{Z} = \mathrm{COMBINE}(\mathbf{H}, \mathbf{U})$;

distortion function $f^{\mathbf{X}}_{Aug}$, and network distortion function $f^{\mathcal{G}}_{Aug}$. Our algorithm is motivated by the empirical results in Section 4 which showed that a NE method for heterophilous networks should have the ability to distinguish nodes with different attributes or structural information. Therefore, we summarise each node's relevant node attribute and structural information into an r-ego network and define the NE task as an r-ego network discrimination problem. We sample a set of r-ego networks $\{\mathcal{G}_r(1), \mathcal{G}_r(2), \ldots, \mathcal{G}_r(n)\}$ from $\mathcal{G}$ (Line 1). Next, we extract network structure of each r-ego network $\mathcal{G}_r(v)$ and enhance each with identity features ($\widetilde{\mathbf{X}}_{id}$) and network structure features ($\widetilde{\mathbf{X}}_{struc}$) to obtain $\widetilde{\mathcal{G}}_r(v) = \{\mathcal{N}^r_v, \mathcal{E}^r_v, \widetilde{\mathbf{X}}\}$ (Line 2). Then, we generate distorted node attribute matrix instances $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ and two distorted instances $(\widetilde{\mathcal{G}}^{(1)}_r(v), \widetilde{\mathcal{G}}^{(2)}_r(v))$ for each r-ego network $\widetilde{\mathcal{G}}_r(v)$ (Line 3-4). For each node $v \in \mathcal{G}$, we compute its node attribute embedding ($\mathbf{H}_v$) and distorted node attribute embeddings ($\mathbf{H}^{(1)}_v$, $\mathbf{H}^{(2)}_v$), and network structure embeddings ($\mathbf{U}^{(1)}_v$, $\mathbf{U}^{(2)}_v$) (Line 8-10). Then, Barlow-Twins and network attribute reconstruction loss functions are employed to compute the final loss $\mathcal{L}$ (Line 11). We update SELENE's parameters (θ, δ) by descending the gradient $\nabla_{\theta,\delta}\mathcal{L}$ (Line 13) until convergence. After the model is trained, we compute node attribute embedding $\mathbf{H}_v$ and network structure embedding $\mathbf{U}_v$ (Line 15-18) to obtain the final node representations $\mathbf{Z}$ by combining $\mathbf{H}$ and $\mathbf{U}$ (Line 19).

6 Evaluation

6.1 Real-world and Synthetic Datasets

Real-world datasets. We use a total of 12 real-world datasets (Texas (Pei et al., 2020), Wisconsin (Pei et al., 2020), Actor (Pei et al., 2020), Chameleon (Rozemberczki et al., 2021), USA-Airports (Ribeiro et al.,

Table 1: Statistics of real-world datasets. $|\mathcal{V}|$: number of nodes; $|\mathcal{E}|$: number of edges; π: dimensionality of nodes features; OSF: nodes only have structure related features; d_{avg}: average degree; $|\mathcal{Y}|$: number of possible class labels; h: homophily ratio;

Dataset	$\|\mathcal{V}\|$	$\|\mathcal{E}\|$	π	OSF	$\|\mathcal{Y}\|$	d_{avg}	h
Texas	183	325	1,703	False	5	1.8	0.108
Wisconsin	251	515	1,703	False	5	2.1	0.196
Actor	7,600	30,019	932	False	5	3.9	0.219
Chameleon	2,277	31,421	2,325	False	5	27.6	0.233
USA-Airports	1,190	13,599	1	True	4	22.9	0.251
Cornell	183	298	1,703	False	5	1.6	0.305
Europe-Airports	399	11,988	1	True	4	30.1	0.309
Brazil-Airports	131	2,077	1	True	4	16.4	0.311
Deezer-Europe	28,281	185,504	31,241	False	2	6.6	0.525
Citeseer	3,327	4,552	3,703	False	6	2.7	0.736
DBLP	4,057	3,528	334	False	4	1.7	0.799
Pubmed	19,717	88,648	500	False	3	4.5	0.802

2017), Cornell (Pei et al., 2020), Europe-Airports (Ribeiro et al., 2017), Brazil-Airports (Ribeiro et al., 2017), Deezer-Europe (Rozemberczki & Sarkar, 2020), Citeseer (Kipf & Welling, 2017), DBLP (Fu et al., 2020), Pubmed (Kipf & Welling, 2017)) in diverse domains (web-page, citation, co-author, flight transport and online user relation). All real-datasets are available online[2]. Statistics information is summarised in Table 1.

Synthetic datasets. Moreover, we generate random synthetic networks with various homophily ratios h by adopting a similar approach to Abu-El-Haija et al. (2019); Kim & Oh (2021). Here, we give detailed descriptions of the generation process.

Graph generation. We generate synthetic graph $\mathcal{G}$ of $|\mathcal{V}|$ nodes with $|\mathcal{Y}|$ different class labels, and $\mathcal{G}$ has $|\mathcal{V}|/|\mathcal{Y}|$ nodes per class. $|\mathcal{V}|$ and $|\mathcal{Y}|$ are two prescribed numbers to determine the size of $\mathcal{G}$. A synthetic graph's homophily ratio h is mainly controlled by p_{in} and p_{out}, where p_{in} means the possibility of existing an edge between two nodes with the same label and p_{out} is the possibility of existing an edge between two nodes with different class labels. Furthermore, the average degree of $\mathcal{G}$ is $d_{avg} = |\mathcal{V}|/|\mathcal{Y}| \cdot \delta$, where $\delta = p_{in} + (|\mathcal{Y}| - 1) \cdot p_{out}$. Following the described graph generation process, with given $|\mathcal{V}|$, $\mathcal{Y}$ and d_{avg}, we choose p_{in} from $\{0.0001\delta, 0.1\delta, 0.2\delta, \ldots, 0.9\delta\}$. Note that the synthetic graph generation process requires both p_{in} and p_{out} are positive numbers, hence we use $p_{in} = 0.0001\delta$ to estimate $h = 0$.

6.2 Experimental Setup

We compare our framework SELENE with 11 competing NE methods in terms of the challenging node clustering task. We adopt 2 different competing NE methods without NNs, including node2vec (N2V) (Grover & Leskovec, 2016) and struc2vec (S2V) (Ribeiro et al., 2017). We adopt 9 additional competing NE methods using NNs, including AE (Hinton & Salakhutdinov, 2006), GAE (Kipf & Welling, 2016), GraphSAGE (SAGE) (Hamilton et al., 2017), DGI (Velickovic et al., 2019), SDCN (Bo et al., 2020), GMI (Peng et al., 2020), GBT (Bielak et al., 2021), H2GCN (Zhu et al., 2020) and FAGCN (Bo et al., 2021). Note that GBT is an SSL approach that applies the Barlow-Twins (Zbontar et al., 2021) strategy to network-structured data. Albeit providing a new model training strategy, the basic GNN building blocks remain the same, hence still maintaining a homophily assumption. H2GCN (Zhu et al., 2020) and FAGCN (Bo et al., 2021) are state-of-the-art (SOTA) heterophilous GNN operators for supervised settings. Here, we train them using the same mechanism as GBT to adapt them to the unsupervised setting. We rename the unsupervised adaptations of H2GCN and FAGCN as H2GCN* and FAGCN*, respectively. See Appendix B and Appendix C for details of all competing methods and implementation.

[2]https://pytorch-geometric.readthedocs.io/en/latest/modules/datasets.html

Table 2: Node clustering results on real-world datasets. The **bold** and underline numbers represent the top-2 results. OOM: out-of-memory.

Dataset	Metrics	AE	N2V	S2V	GAE	SAGE	SDCN	DGI	GMI	GBT	H2GCN*	FAGCN*	Ours	↑ (%)
							Heterophilous datasets							
Texas $h = 0.11$	ACC	50.49	48.80	49.73	42.02	56.83	44.04	55.74	35.19	55.46	58.80	57.92	**65.23**	10.94
	NMI	16.63	2.58	18.61	8.49	16.97	14.24	8.73	7.72	10.17	22.49	23.35	**25.40**	8.78
	ARI	14.6	-1.62	20.97	10.83	23.50	10.65	8.25	2.96	12.10	25.04	22.54	**34.21**	36.62
Wisc. $h = 0.20$	ACC	58.61	41.39	43.03	37.81	46.29	38.25	44.58	36.97	48.01	64.18	61.91	**71.69**	11.70
	NMI	30.92	4.23	11.23	9.19	10.16	8.46	10.72	11.68	7.55	29.64	27.35	**39.51**	27.78
	ARI	28.53	-0.48	11.50	5.2	6.06	3.67	10.31	3.74	3.85	32.61	31.56	**43.48**	33.33
Actor $h = 0.22$	ACC	24.19	25.02	22.49	23.45	23.08	23.67	24.26	26.18	24.68	25.55	25.61	**29.03**	13.35
	NMI	0.97	0.09	0.04	0.18	0.58	0.08	1.38	0.20	0.74	3.23	3.22	**4.72**	46.13
	ARI	0.50	0.06	-0.05	-0.04	0.22	-0.01	0.07	0.41	-0.57	0.31	0.34	**1.84**	268.00
Chamel. $h = 0.23$	ACC	35.68	21.31	26.34	32.76	31.04	33.5	27.77	25.73	32.21	30.62	31.33	**38.97**	9.22
	NMI	10.38	0.34	3.55	11.60	10.55	9.57	4.42	2.5	10.56	14.62	14.71	**20.63**	40.24
	ARI	5.80	0.02	1.82	4.65	6.16	5.86	1.85	0.52	7.01	4.78	5.16	**15.94**	127.39
USA-Air. $h = 0.25$	ACC	55.24	26.29	27.58	30.84	32.96	33.52	33.36	28.69	34.96	39.01	38.82	**58.90**	6.63
	NMI	30.13	0.25	0.44	2.71	2.67	5.21	5.52	0.6	5.27	12.43	12.30	**31.17**	3.45
	ARI	24.20	-0.05	0.09	2.67	2.52	1.93	4.95	0.29	3.42	9.48	9.33	**25.53**	5.49
Cornell $h = 0.31$	ACC	52.19	50.98	32.68	43.72	44.7	36.94	44.1	33.55	52.19	54.97	56.23	**57.96**	3.08
	NMI	17.08	5.84	1.54	5.11	4.33	6.6	5.79	5.26	5.94	17.05	17.08	**17.32**	1.41
	ARI	17.41	0.18	-2.20	6.51	5.64	3.38	4.87	3.05	0.63	19.50	19.88	**23.03**	15.85
Eu.-Air. $h = 0.31$	ACC	55.36	30.78	36.89	34.84	31.75	37.37	35.59	35.34	39.75	37.27	42.11	**57.80**	4.41
	NMI	32.44	3.69	6.15	10.15	2.10	8.45	10.77	11.08	9.44	9.08	16.81	**34.25**	5.58
	ARI	24.24	0.83	4.49	7.37	1.16	5.31	8.44	8.18	7.87	5.3	11.98	**25.69**	5.98
Bra.-Air. $h = 0.31$	ACC	71.68	30.38	38.93	36.64	37.02	38.7	37.1	38.93	40.92	43.97	44.2	**79.12**	10.38
	NMI	49.26	2.5	10.23	10.96	6.89	14.05	10.64	12.62	12.16	22.37	22.67	**55.90**	13.48
	ARI	42.93	-0.22	5.45	6.56	4.18	7.27	7.02	9.11	8.31	13.99	14.4	**53.21**	23.95
Deezer. $h = 0.53$	ACC	55.88	52.97	OOM	51.51	51.06	54.76	53.16	OOM	OOM	56.81	56.81	**59.77**	5.21
	NMI	0.28	0.0	OOM	0.13	0.16	0.17	0.05	OOM	OOM	0.27	0.27	**0.32**	14.28
	ARI	0.81	0.02	OOM	0.07	-0.02	0.61	-0.23	OOM	OOM	1.22	0.82	**0.89**	8.54
							Homophilous datasets							
Citeseer $h = 0.74$	ACC	58.79	20.76	21.22	48.37	49.28	59.86	58.94	59.04	57.21	47.33	47.42	**60.02**	0.27
	NMI	30.91	0.35	1.18	24.59	22.97	30.37	32.6	32.11	31.9	20.48	20.18	**32.74**	0.31
	ARI	30.29	-0.01	0.17	19.50	19.21	29.7	33.16	33.09	33.17	18.03	17.93	**33.46**	0.87
DBLP $h = 0.80$	ACC	48.50	29.19	31.65	57.81	48.68	61.94	58.22	63.28	73.10	41.87	41.25	**75.74**	3.61
	NMI	18.98	0.14	1.33	28.94	16.46	27.13	29.98	33.91	42.21	11.04	10.60	**44.17**	4.64
	ARI	15.15	-0.04	1.39	18.78	13.38	27.77	26.81	28.77	42.57	4.67	4.32	**46.35**	8.88
Pubmed $h = 0.80$	ACC	65.34	39.32	37.39	42.08	67.66	61.9	65.47	OOM	OOM	56.71	56.88	**67.98**	0.47
	NMI	26.89	0.02	0.07	1.28	30.71	19.71	28.05	OOM	OOM	17.15	16.91	**31.14**	1.40
	ARI	25.98	0.09	0.06	0.15	29.10	18.63	27.25	OOM	OOM	16.51	16.27	**29.79**	2.37

6.3 Experimental Results

Real-world datasets. Node clustering results on homophilous and heterophilous real-world datasets are summarised in Table 2 where we see that SELENE is the best-performing method in all heterophilous datasets. In particular, compared to the best results of competing models, our framework achieves a significant improvement of up to 13.35% on ACC, 46.13% on NMI and 268% on ARI. Such outstanding performance gain demonstrates that SELENE successfully integrates the important node attributes and network structure information into node representations. An interesting case is the comparison of SELENE, H2GCN*, FAGCN* and GBT, given that H2GCN*, FAGCN* and GBT also utilise the Barlow-Twins objective function to optimise a GNN model, and the major difference between H2GCN*, FAGCN*, GBT and SELENE is our designs to address **RC1** and **RC2**. SELENE has a superior performance in all heterophilous datasets, indicating the effectiveness of designs and showing that simply porting supervised models to unsupervised scenarios is not appropriate.

Results of node clustering on homophilous networks, contained in Table 2, show that SELENE achieves new SOTA performance. This demonstrates SELENE's suitability for homophilous networks and proves the flexibility of our designs.

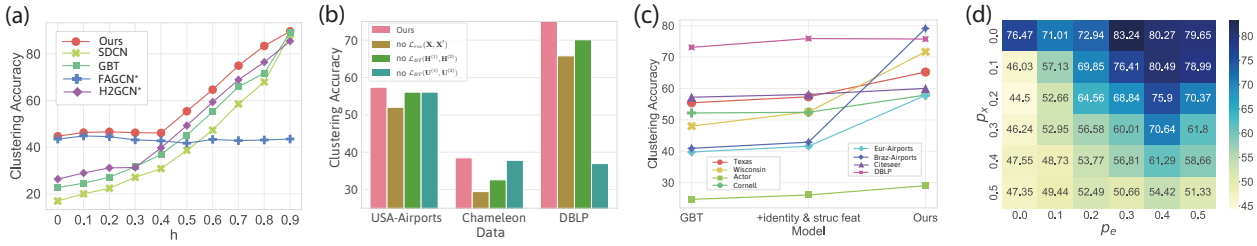

Figure 5: (a) Clustering accuracy comparison on synthetic networks. (b) Loss function ablation on real-world networks. (c) Framework component exploration on real-world networks. (d) Hyperparameter influences exploration on synthetic-0.8.

Synthetic networks. We present SELENE vs SDCN (SOTA node clustering model) vs GBT (SOTA graph contrastive learning model) vs FAGCN* (SOTA heterophilous GNN model) clustering accuracy in Figure 5-(a). SELENE achieves the best performance on the synthetic datasets, which shows SELENE adapts well to homophily/heterophily scenarios with(out) contextually raw node attributes. Moreover, FAGCN* performs worse on 10 synthetic networks, which indicates that the heterophilous GNNs models designed for supervised settings do not adapt well to unsupervised settings because they need supervision information to train the more complex aggregation mechanism. In addition, we observe that SELENE has no significant improvement with $h \leq 0.4$, because network structure encoder's expressive power on extreme homophily is limited.

6.4 Analysis

Model scalability. The network structure encoder module is categorised as a local network algorithm (Teng, 2016), which only involves local exploration of the network structure. On the other hand, the node attribute encoder module naturally supports the mini-batch mechanism. Therefore, our design enables SELENE to scale to representation learning on large-scale networks and to be friendly to distributed computing settings (Qiu et al., 2020). Table 2 illustrates that three competing methods, i.e., struc2vec, GMI and GBT, have out-of-memory issues on large datasets, i.e., Pubmed and Deezer., an issue which did not arise with SELENE. This shows SELENE's advantage in handling large scale networks due to its local network algorithm characteristic.

Effectiveness of loss function. The objective loss function of SELENE contains three components, and we thus sought to test the effectiveness of each component. In particular, we ablate each component and evaluate the obtained node representations on two heterophilous datasets (USA-Air., Chamel.) and one homophilous dataset (DBLP). Results are shown in Figure 5-(b) indicate that ablation of any component decreases the model's performance. Specifically, the ablation of $\mathcal{L}_{Rec}(\mathbf{X}, \widehat{\mathbf{X}})$ causes steeper performance degradation in heterophilous datasets, and the ablation of $\mathcal{L}_{BT}(\mathbf{U}^{(1)}, \mathbf{U}^{(2)})$ causes steeper performance degradation in homophilous datasets, which indicates the importance of node attributes and network structure information for heterophilous and homophilous networks, respectively (consistent with observations in Section 4).

Effectiveness of dual-channel feature embedding pipeline. SELENE contains a novel dual-channel features embedding pipeline to integrate node attributes and network structure information, thus, we conduct an ablation study to explore the effectiveness of this pipeline. We first remove the pipeline and only use the Barlow-Twins loss function to train a vanilla GCN encoding module (such a structure is the same as GBT, hence we remark it as GBT). Next, we add the network structure channel, which includes r-ego network extraction, anonymisation and distortion. Lastly, we add the node attribute channel to form the complete SELENE framework. Experimental results are shown in Figure 5-(c). Overall, we observe that the design of each channel is useful for learning better representation, with the node attribute channel playing a major role in the embedding of heterophilous networks. Note that adding the node attribute channel slightly decreases the clustering accuracy for the homophilous dataset, i.e., DBLP, but it is still competitive.

Influence of p_x and p_e. We present SELENE's clustering accuracy on synthetic-0.8 with different p_x and p_e in Figure 5-(d). The figure indicates that hyperparameters of distortion methods significantly influence representation quality.

Table 3: Node classification results on real-world datasets. The **bold** and underline numbers represent the top-2 results.

Dataset	F1-score	AE	N2V	S2V	GAE	SAGE	SDCN	DGI	GMI	GBT	H2GCN*	FAGCN*	Ours	↑ (%)
Heterophilous datasets														
Texas	Micro	0.598	0.509	0.592	0.5	0.563	0.552	0.541	0.505	0.461	0.523	0.607	**0.643**	5.93
	Macro	0.284	0.165	0.335	0.253	0.301	0.153	0.266	0.295	0.253	0.186	0.265	**0.381**	13.73
Actor	Micro	0.284	0.240	0.227	0.254	0.277	0.252	0.272	0.272	0.245	0.244	0.281	**0.341**	20.07
	Macro	0.194	0.178	0.190	0.164	0.204	0.125	0.244	0.195	0.224	0.138	0.166	**0.274**	12.30
USA-Air.	Micro	0.541	0.239	0.254	0.309	0.311	0.302	0.338	0.253	0.35	0.535	0.527	**0.565**	4.44
	Macro	0.476	0.235	0.253	0.236	0.299	0.288	0.260	0.221	0.288	0.483	0.454	**0.532**	10.14
Bra.-Air.	Micro	0.611	0.247	0.301	0.331	0.312	0.304	0.321	0.329	0.371	0.465	0.507	**0.715**	17.02
	Macro	0.538	0.203	0.284	0.244	0.295	0.281	0.244	0.249	0.319	0.415	0.420	**0.692**	28.62
Homophilous datasets														
DBLP	Micro	0.746	0.269	0.334	0.768	0.732	0.638	0.792	0.811	0.744	0.413	0.560	**0.813**	0.25
	Macro	0.737	0.227	0.305	0.758	0.721	0.633	0.784	0.804	0.736	0.371	0.536	**0.810**	0.75

Table 4: Link prediction results on real-world datasets. The **bold** and underline numbers represent the top-2 results.

Dataset	Metric	AE	N2V	S2V	GAE	SAGE	SDCN	DGI	GMI	GBT	H2GCN*	FAGCN*	Ours
Heterophilous datasets													
Texas	ROC-AUC	0.529	0.549	0.571	0.634	0.559	0.640	**0.787**	0.777	0.783	0.451	0.673	0.784
Actor	ROC-AUC	0.501	0.673	0.652	0.668	0.634	0.552	0.626	0.609	0.598	0.679	0.613	**0.725**
USA-Air.	ROC-AUC	0.518	0.714	0.685	0.842	0.595	0.528	0.857	0.854	0.867	0.667	0.642	**0.871**
Bra.-Air.	ROC-AUC	0.665	0.592	0.637	0.787	0.615	0.677	0.855	0.791	0.786	0.702	0.696	**0.881**
Homophilous datasets													
DBLP	ROC-AUC	0.872	0.758	0.717	0.870	0.841	0.628	**0.939**	0.884	0.892	0.736	0.764	0.908

Additional experiments. In Section 6.3, we presented node clustering results of various models. Meanwhile, we acknowledge that some works adopt a different experiment pipeline that utilises node labels to train a classifier, after obtaining unsupervised node representations, to predict labels of test nodes (Velickovic et al., 2019; Peng et al., 2020; Bielak et al., 2021). We argue that this is not a precise setting because if some node labels are available to train a classifier, we can directly utilise them to train the NE model (the typical supervised setting). However, hereby we still provide results of a number of real-world datasets that follow this setting to show the generality of SELENE. Results in Table 3 show that SELENE achieves the best classification performance on all datasets with homophily and heterophily. Moreover, we also report another major graph analysis task, i.e., link prediction, performances of a number of real-world dataset in Table 4. From the experimental results, we can find that *(i)* homophily ratio has significant influence on link prediction task, and *(ii)* SELENE performs priority on this task as well. More experimental settings and experimental results discussion refer to Appendix D.

7 Conclusion and Future Directions

In this paper, we focused on the unsupervised network embedding task with challenging heterophily settings and tackled two main research questions. First, we showed through an empirical investigation that the performance of existing embedding methods that utilise network structure decreases significantly with the decrease of network homophily ratio. Second, to address the identified limitations, we proposed SELENE, which effectively fuses node attributes and network structure information without additional supervision. Comprehensive experiments demonstrated the significant performance of SELENE, and additional ablation analysis confirms the effectiveness of components of SELENE on real-world and synthetic networks. As future work, *(i)* we propose to explore the *Information Bottleneck* (Tishby & Zaslavsky, 2015; Wu et al., 2020) of network embedding to theoretically define the optimal representation of an arbitrary network (Zügner et al., 2018); *(ii)* designing a more powerful unsupervised network structure encoder for extreme heterophilous networks is also a promising future work.

References

Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *Proceedings of the 2019 International Conference on Machine Learning (ICML)*, pp. 21–29. JMLR, 2019.

Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinícius Flores Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Çaglar Gülçehre, H. Francis Song, Andrew J. Ballard, Justin Gilmer, George E. Dahl, Ashish Vaswani, Kelsey R. Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matthew Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. *CoRR*, abs/1806.01261, 2018.

Piotr Bielak, Tomasz Kajdanowicz, and Nitesh V. Chawla. Graph barlow twins: A self-supervised representation learning framework for graphs. *CoRR*, abs/2106.02466, 2021.

Deyu Bo, Xiao Wang, Chuan Shi, Meiqi Zhu, Emiao Lu, and Peng Cui. Structural deep clustering network. In *Proceedings of the 2020 International Conference on World Wide Web (WWW)*, pp. 1400–1410. ACM, 2020.

Deyu Bo, Xiao Wang, Chuan Shi, and Huawei Shen. Beyond low-frequency information in graph convolutional networks. In *Proceedings of the 2021 AAAI Conference on Artificial Intelligence (AAAI)*, pp. 3950–3957. AAAI, 2021.

Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proceedings of the 2020 AAAI Conference on Artificial Intelligence (AAAI)*, pp. 3438–3445. AAAI, 2020.

Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering*, 31(5):833–852, 2019.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of the 2016 Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 3837–3845. NIPS, 2016.

Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. MAGNN: metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of the 2020 International Conference on World Wide Web (WWW)*, pp. 2331–2341. ACM, 2020.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 2016 ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 855–864. ACM, 2016.

William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 2017 Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 1025–1035. NIPS, 2017.

Jun Han and Claudio Moraga. The influence of the sigmoid function parameters on the speed of backpropagation learning. In *Proceedings of the 1995 International Workshop on Artificial Neural Networks (IWANN)*, pp. 195–201. Springer, 1995.

John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034. IEEE, 2015.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313:504–507, 2006.

Dongkwan Kim and Alice Oh. How to find your friendly neighborhood: Graph attention design with self-supervision. In *Proceedings of the 2021 International Conference on Learning Representations (ICLR)*. OpenReview.net, 2021.

Thomas N. Kipf and Max Welling. Variational graph auto-encoders. *CoRR*, abs/1611.07308, 2016.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 2017 International Conference on Learning Representations (ICLR)*. OpenReview.net, 2017.

Ron Levie, Federico Monti, Xavier Bresson, and Michael M. Bronstein. Cayleynets: Graph convolutional neural networks with complex rational spectral filters. *IEEE Signal Processing Magazine*, 67(1):97–109, 2017.

Michelle M. Li, Kexin Huang, and Marinka Zitnik. Adaptive multi-grained graph neural networks. *CoRR*, abs/2104.04883, 2021.

Pan Li, Yanbang Wang, Hongwei Wang, and Jure Leskovec. Distance encoding - design provably more powerful graph neural networks for structural representation learning. In *Proceedings of the 2020 Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2020.

Derek Lim, Felix Hohne, Xiuyu Li, Sijia Linda Huang, Vaishnavi Gupta, Omkar Bhalerao, and Ser-Nam Lim. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. In *Proceedings of the 2021 Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2021.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian J. Goodfellow. Lstm-based encoder-decoder for multi-sensor anomaly detection. *CoRR*, abs/1511.05644, 2015.

Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. Lstm-based encoder-decoder for multi-sensor anomaly detection. *CoRR*, abs/1607.00148, 2016.

Jonathan Masci, Ueli Meier, Dan C. Ciresan, and Jürgen Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. In *Proceedings of the 2011 International Conference on Artificial Neural Networks (ICANN)*, pp. 52–59. Springer, 2011.

Julian J. McAuley and Jure Leskovec. Learning to discover social circles in ego networks. In *Proceedings of the 2012 Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 548–556. NIPS, 2012.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionally. In *Proceedings of the 2013 Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 3111–3119. NIPS, 2013.

Yimeng Min, Frederik Wenkel, and Guy Wolf. Scattering GCN: overcoming oversmoothness in graph convolutional networks. In *Proceedings of the 2020 Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2020.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 2010 International Conference on Machine Learning (ICML)*, pp. 807–814. JMLR, 2010.

Shashank Pandit, Duen Horng Chau, Samuel Wang, and Christos Faloutsos. Netprobe: a fast and scalable system for fraud detection in online auction networks. In *Proceedings of the 2007 International Conference on World Wide Web (WWW)*, pp. 201–210. ACM, 2007.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *Proceedings of the 2020 International Conference on Learning Representations (ICLR)*. OpenReview.net, 2020.

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 2020 International Conference on World Wide Web (WWW)*, pp. 259–270. ACM, 2020.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 2014 ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 701–710. ACM, 2014.

Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *Proceedings of the 2018 ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 459–467. ACM, 2018.

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 2020 ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 1150–1160. ACM, 2020.

Leonardo Filipe Rodrigues Ribeiro, Pedro H. P. Saverese, and Daniel R. Figueiredo. *struc2vec*: Learning node representations from structural identity. In *Proceedings of the 2017 ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 385–394. ACM, 2017.

Benedek Rozemberczki and Rik Sarkar. Characteristic functions on graphs: Birds of a feather, from statistical descriptors to parametric models. In *Proceedings of the 2020 ACM International Conference on Information and Knowledge Management (CIKM)*, pp. 1325–1334. ACM, 2020.

Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2), 2021.

Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 2015 International Conference on World Wide Web (WWW)*, pp. 1067–1077. ACM, 2015.

Jie Tang, Jimeng Sun, Chi Wang, and Zi Yang. Social influence analysis in large-scale networks. In *Proceedings of the 2009 ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 807–816. ACM, 2009.

Shang-Hua Teng. Scalable algorithms for data and network analysis. *Foundations and Trends in Theoretical Computer Science*, 12(1-2):1–274, 2016.

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pp. 1–5. IEEE, 2015.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *Proceedings of the 2018 International Conference on Learning Representations (ICLR)*. OpenReview.net, 2018.

Petar Velickovic, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R. Devon Hjelm. Deep graph infomax. In *Proceedings of the 2019 International Conference on Learning Representations (ICLR)*. OpenReview.net, 2019.

Tailin Wu, Hongyu Ren, Pan Li, and Jure Leskovec. Graph information bottleneck. In *Proceedings of the 2020 Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2020.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1): 4–24, 2021.

Feng Xia, Ke Sun, Shuo Yu, Abdul Aziz, Liangtian Wan, Shirui Pan, and Huan Liu. Graph learning: A survey. *IEEE Transactions on Artificial Intelligence*, 2(2):109–127, 2021.

Bingbing Xu, Huawei Shen, Qi Cao, Yunqi Qiu, and Xueqi Cheng. Graph wavelet neural network. In *Proceedings of the 2019 International Conference on Learning Representations (ICLR)*. OpenReview.net, 2019a.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *Proceedings of the 2019 International Conference on Machine Learning (ICML)*. JMLR, 2019b.

Zhen Yang, Ming Ding, Chang Zhou, Hongxia Yang, Jingren Zhou, and Jie Tang. Understanding negative sampling in graph representation learning. In *Proceedings of the 2020 ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 1666–1676. ACM, 2020.

Jiaxuan You, Jonathan Michael Gomes Selman, Rex Ying, and Jure Leskovec. Identity-aware graph neural networks. In *Proceedings of the 2021 AAAI Conference on Artificial Intelligence (AAAI)*, pp. 10737–10745. AAAI, 2021.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In *Proceedings of the 2020 Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2020.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *Proceedings of the 2021 International Conference on Machine Learning (ICML)*, pp. 12310–12320. JMLR, 2021.

Ziwei Zhang, Peng Cui, and Wenwu Zhu. Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

Xin Zheng, Yixin Liu, Shirui Pan, Miao Zhang, Di Jin, and Yu Philip S. Graph neural networks for graphs with heterophily: A survey. *CoRR*, abs/2202.07082, 2022.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. In *Proceedings of the 2020 Annual Conference on Neural Information Processing Systems (NeurIPS)*. NeurIPS, 2020.

Jiong Zhu, Ryan A. Rossi, Anup B. Rao, Tung Mai, Nedim Lipka, Nesreen K. Ahmed, and Danai Koutra. Graph neural networks with heterophily. In *Proceedings of the 2021 AAAI Conference on Artificial Intelligence (AAAI)*, pp. 11168–11176. AAAI, 2021.

Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. In *Proceedings of the 2018 ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 2847–2856. ACM, 2018.

A Real-world Dataset Description and Evaluation Metrics

In our experiments, we use the following real-world datasets with varying homophily ratios h. Their statistics are provided in Table 1.

- **Texas**, **Wisconsin** and **Cornell** (Zhu et al., 2020) are networks that represent links between web pages of the corresponding universities, originally collected by the CMU WebKB project. Nodes of these networks are web pages, which are classified into 5 categories: course, faculty, student, project and staff.
- **Chameleon** (Zhu et al., 2020) is a subgraph of web pages in Wikipedia discussing the corresponding topics
- **Actor** (Zhu et al., 2020) is the actor-only reduced subgraph of the film-director-actor-writer network (Tang et al., 2009). Each node corresponds to an actor, node features are keywords of the actor's Wikipedia page, and edges mean whether two actors denote co-occurrence on the same page. Nodes are classified into five categories in terms of words from actors' Wikipedia.
- **USA-Airports**, **Europe-Airports**, **Brazil-Airports** are three air traffic networks collected from government websites throughout the year 2016 and were used to evaluate algorithms to learn structure representations of nodes. Nodes represent airports, and edges indicate whether there are commercial flights between them. The goal is to infer the level of an airport using the connectivity pattern solely.
- **Deezer-Europe** (Zheng et al., 2022) is a social network of users on Deezer from European countries, where edges represent mutual follower relationships. The node features are based on artists liked by each user. Nodes are labelled with reported gender.
- **Citeseer** and **Pubmed** are papers citation networks that were originally introduced in Kipf & Welling (2017), which contains sparse bag-of-words feature vectors for each document and a list of citation links between these documents.
- **DBLP** (Bo et al., 2020) is an author network from the DBLP dataset. There is an edge between the two authors if they have a co-author relationship in the dataset. And the authors are divided into four research areas: database, data mining, artificial intelligence, and computer vision. We label each author's research area depending on the conferences they submitted. Author features are the elements of a bag of words represented by keywords.

Evaluation metrics. We employ three node clustering evaluation metrics: accuracy (ACC), normalised mutual information (NMI) and average rand index (ARI). For each evaluation metric, a higher value means better node clustering performance.

B Competing Methods Description

- **AE** (Hinton & Salakhutdinov, 2006): A classical deep representation learning method relies on raw node attributes with an encoder-decoder pipeline.
- **node2vec** (Grover & Leskovec, 2016): A random-walk based shallow node representation learning method with two hyper-parameter to adjust the random walk.
- **struc2vec** (Ribeiro et al., 2017): A shallow node representation learning method that learns node representations by identifying node's related local structure and estimating their relationship to other nodes. It relies on network structure information.
- **GAE** (Kipf & Welling, 2016): Deep node representation learning methods using GCN to learn node representations by reconstructing the network structure.
- **GraphSAGE** (Hamilton et al., 2017): A deep node representation learning method with a specifically designed loss function that makes nodes far apart have different representations. It relies on both node attributes and network structure.

- **DGI** (Velickovic et al., 2019): A self-supervised learning model by migrating infomax techniques from computer vision domain to networks.

- **GMI** (Peng et al., 2020): It generalises the idea of conventional mutual information computations from vector space to the graph domain where measuring mutual information from two aspects of node features and topological structure is indispensable.

- **SDCN** (Bo et al., 2020): Deep learning node representations methods specifically for node clustering task. An AE component relies on raw node attributes, and another GCN component merges structure and node attributes together and trains with a dual self-supervised module.

- **GBT** (Bielak et al., 2021): A self-supervised learning approach that migrates the Barlow-Twins approach to network structure data. However, they don't modify the model structure of GNNs but just provide a new mode training approach, hence still maintaining the homophily assumption.

- **H2GCN** (Zhu et al., 2020): A heterophilous GNN operator that separately aggregates information from ego node, neighbouring nodes and high-order neighbouring nodes under supervised settings.

- **FAGCN** (Bo et al., 2021): Different from conventional GNN operators that only have a low-frequency filter, they propose another high-frequency filter to collaboratively work with low-frequency filter to work on networks with homophily and heterophily, under supervised settings.

C Experimental Setup & Hyperparameter Tuning

For network embedding methods without neural networks, we set the embedding dimension to 128, the number of random walks of each node to 10 and the walk length to 80. For node2vec, we additionally select p, q over $\{0.25, 0.5, 1, 2\}$ with best clustering performance. We utilise the integrated implementations from GraphEmbedding[3].

We train network embedding methods with neural networks, including AE, GAE, VGAE and SDCN, with the same settings as Bo et al. (2020). Specifically, we train the models end-to-end using all nodes and edges with 30 epochs and a learning rate of 10^{-3}. For AE, we set the representation dimensions to $\{\pi - 500 - 500 - 200 - 10\}$, where π is the dimensionality of raw node attributes. For GNN-related methods, including GAE, VGAE, GraphSAGE, DGI, GBT, GMI, FAGCN* and SELENE, we set their representation dimensions to $\{\pi - 256 - 16\}$. The representation dimensions of SDCN's GCN module are the same as AE (Bo et al., 2020) proposes to pretrain the SDCN's AE component to boost its performance, thus, we report the best performance of SDCN with/without pre-trained AE. For DGI, GMI, GBT and SELENE, we assign the same GCN (Kipf & Welling, 2017) encoder and follow the optimisation protocol as Bielak et al. (2021). We set $r = 3$ for ego network extraction following Li et al. (2020) and the batch size $= 512$.

For GAE, VGAE, GraphSAGE, DGI and FAGCN* we use the implementation from Pytorch-Geometric[4]; for AE & SDCN and GMI, GBT and H2GCN*, we use the implementation from the published code of SDCN[5], GMI[6], GBT[7] and and H2GCN*[8], respectively,

Note that, all experiments are conducted on a single Tesla V100 GPU. For all NN-based methods, we initialise them 10 times with random seeds and select the best solution to follow the similar setting as Bo et al. (2020). After obtaining node representations with each model, we feed the learned representations into a K-means clustering model (Hartigan & Wong, 1979) to get the final clustering prediction. The final clustering section is thus repeated 10 times, and we report the mean/std performance.

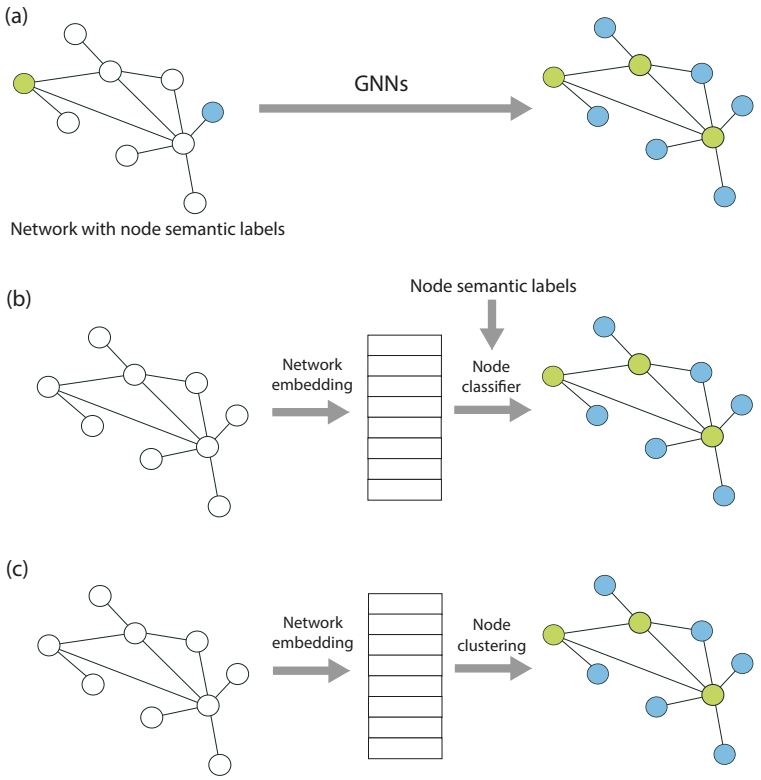

Figure 6: Comparison of pipelines for node classification. (a) In (semi-)supervised node classification with GNNs, network structure and node semantic labels of nodes in the training set are provided to the model; (b) Network embedding obtains low-dimensional representations for nodes without using semantic labels, then a classifier is trained in a semi-supervised fashion using labels of nodes in the training set; (c) Most challenging setting in which both network embedding and prediction of node semantic labels do not use supervision from labels in the training set.

D Additional Experimental Results and Discussion

This paper focuses on the unexplored and challenging research task: unsupervised network embedding with heterophily. Therefore, in the experimental evaluation section (Section 6), we mainly utilise the node clustering task as the downstream application to evaluate the quality of embeddings generated by different models. As shown in Figure 6-(c), we do not use any supervisory signal during the pipeline. On the contrary, the (semi-)supervised node classification settings (as shown in Figure 6-(a)) takes known node semantic labels to train the model to predict the class label of unknown nodes. Note that we acknowledge that some works adopt a different experiment pipeline that does not use node semantic labels for network embedding but utilises node labels to train a classifier, after obtaining unsupervised node representations, to predict labels of test nodes (Velickovic et al., 2019; Peng et al., 2020; Bielak et al., 2021). We argue that this is not a precise setting because if some node semantic labels are available to train a classifier, we can directly utilise them to train the NE model (the typical supervised setting). Therefore, we mainly focus on the node clustering task strictly following the unsupervised setting. Moreover, we still provide results of a number of real-world datasets (Figure 3) that follow the settings Velickovic et al. (2019); Peng et al. (2020); Bielak et al. (2021) to

[3] https://github.com/shenweichen/GraphEmbedding
[4] https://pytorch-geometric.readthedocs.io/en/latest/
[5] https://github.com/bdy9527/SDCN
[6] https://github.com/zpeng27/GMI
[7] https://github.com/pbielak/graph-barlow-twins
[8] https://github.com/GemsLab/H2GCN

show the generality of SELENE. We employ two node classification evaluation metrics: Micro- and Macro-F1 Scores.

Moreover, we also report the experimental results on another major graph application task, i.e., link prediction, which wants to predict missing links to complete incomplete graph structure. The link prediction experimental settings are similar to node2vec paper (Grover & Leskovec, 2016). Particularly, we randomly remove 20% edges from the graph as positive edges, and set half of these edges as validation positive edges and test positive edges, respectively. We also sample another set of validation and test negative edges, which has the equal number of negative edges (i.e., edges that do not exist in the complete known graph structure) as positive edge set. The rest positive edges are used as training set to train the network embedding model to learn node embeddings, the training mechanism is the same as unsupervised network embedding. In the end, we can predict missing edges by investigating the similarity between a pair of node embeddings. We employ ROC-AUC metric to evaluate the link prediction performance. For each dataset, we randomly generate 10 groups of samplings to test each model and report the average values in Table 4. Results in Table 4 show that *(i)* homophily ratio has significant influence on link prediction task. The performance of network embeddings models on homophilous dataset, i.e., DBLP, is more outstanding than on heterophilous datasets. And *(ii)* SELENE performs priority on this task as well.

E Additional Information on Barlow-Twins Loss Function

Barlow-Twins (Zbontar et al., 2021) is a method that learns data representations using a symmetric network architecture and an empirical cross-correlation based loss function. Specifically, given an input image $\mathbf{X}$ and a convolutional neural network models f_θ. We can compute the image embedding $\mathbf{H} = f_\theta(\mathbf{X})$. Barlow-Twins loss is formally desined as:

$$\mathcal{L}_{BT} \triangleq \underbrace{\sum_i (1 - \mathcal{C}_{ii})^2}_{\text{invariance term}} + \lambda \underbrace{\sum_i \sum_{j \neq i} {\mathcal{C}_{ij}}^2}_{\text{redundancy reduction term}} \tag{7}$$

where λ is a positive constant trading off the importance of the first and second terms of the loss, and where $\mathcal{C}$ is the cross-correlation matrix computed between the outputs of the two identical networks along the batch dimension:

$$\mathcal{C}_{ij} \triangleq \frac{\sum_b \mathbf{H}^A_{b,i} \mathbf{H}^B_{b,j}}{\sqrt{\sum_b \left(\mathbf{H}^A_{b,i}\right)^2} \sqrt{\sum_b \left(\mathbf{H}^B_{b,j}\right)^2}} \tag{8}$$

where b indexes batch samples and i, j index the vector dimension of the networks' outputs. $\mathcal{C}$ is a square matrix with a size of the dimensionality of the network's output and with values comprised between -1 (i.e. perfect anti-correlation) and 1 (i.e. perfect correlation).

Intuitively, the *invariance term* of the objective, by trying to equate the diagonal elements of the cross-correlation matrix to 1, makes the embedding invariant to the distortions applied. The *redundancy reduction term*, by trying to equate the off-diagonal elements of the cross-correlation matrix to 0, decorrelates the different vector components of the embedding. This decorrelation reduces the redundancy between output units so that the output units contain non-redundant information about the sample.