Representation Learning on Heterostructures via Heterogeneous Anonymous Walks

Xuan Guo, Pengfei Jiao, *Member, IEEE*, Wang Zhang, Ting Pan, Mengyu Jia, Danyang Shi, and Wenjun Wang

***Abstract*— Capturing structural similarity has been a hot topic in the field of network embedding (NE) recently due to its great help in understanding node functions and behaviors. However, existing works have paid very much attention to learning structures on homogeneous networks, while the related study on heterogeneous networks is still void. In this article, we try to take the first step for representation learning on heterostructures, which is very challenging due to their highly diverse combinations of node types and underlying structures. To effectively distinguish diverse heterostructures, we first propose a theoretically guaranteed technique called heterogeneous anonymous walk (HAW) and give two more applicable variants. Then, we devise the HAW embedding (HAWE) and its variants in a data-driven manner to circumvent using an extremely large number of possible walks and train embeddings by predicting occurring walks in the neighborhood of each node. Finally, we design and apply extensive and illustrative experiments on synthetic and real-world networks to build a benchmark on heterostructure learning and evaluate the effectiveness of our methods. The results demonstrate our methods achieve outstanding performance compared with both homogeneous and heterogeneous classic methods and can be applied on large-scale networks.**

***Index Terms*— Heterogeneous network, network embedding (NE), role discovery, structural similarity, unsupervised learning.**

NOMENCLATURE

$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	Homogeneous network.
$\mathcal{H} = (\mathcal{V}, \mathcal{E}, \Phi, \Psi)$	Heterogeneous network.
$\mathcal{V}$, $\mathcal{E}$	Node set and edge set of $\mathcal{G}$ or $\mathcal{H}$.
$\mathcal{T}$, $\mathcal{T}_{\mathcal{E}}$	Sets of node types and edge types in $\mathcal{H}$.
$\Phi(v)$, $\Psi(e)$	Node- and edge-type mappings in $\mathcal{H}$.
$\mathcal{N}(v)$	Set of node v's neighbors in $\mathcal{G}$ or $\mathcal{H}$.
$w = (v_0, v_1, \ldots, v_l)$	Random walk with length l.
$a = (f(v_0), f(v_1), \ldots, f(v_l))$	AW transformed from w in $\mathcal{G}$.
$f(v)$	Node position function over w.
$\mathcal{G}_r(v)$	r-hop local subgraph of v in $\mathcal{G}$.
$h = (g(v_0), g(v_1), \ldots, g(v_l))$	HAW transformed from w in $\mathcal{H}$.
$g(v) = (f(v), \Phi(v))$	Function combining f and Φ.
$\mathcal{H}_r(v)$	r-hop local subgraph of v in $\mathcal{H}$.
$C^v = (h_1^v, h_2^v, \ldots, h_T^v)$	Context of v with T words (HAWs).
$\{C^v\}_{v \in \mathcal{V}}$	Heterostructure corpus of $\mathcal{H}$.
$\mathcal{L}$	Lexicon over the corpus $\{C^v\}_{v \in \mathcal{V}}$.
$\mathbf{w}_h \in \mathbb{R}^d$	d-dimensional embedding of HAW h.
$\mathbf{z}_v \in \mathbb{R}^d$	d-dimensional embedding of node v.
$\mathrm{OC}(h)$	Ordered node type counting in h.

Manuscript received 28 April 2022; revised 30 October 2022; accepted 27 December 2022. This work was supported in part by the Intelligent Manufacturing Special Foundation of Tianjin, China, under Grant 20201198; in part by the Key Research and Development Program of Tianjin under Grant 20YFZCSN01010; in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LDT23F01015F01 and Grant LDT23F01012F01; and in part by the Zhejiang Laboratory Open Research Project under Grant K2022QA0AB01. *(Corresponding author: Wenjun Wang.)*

Xuan Guo, Wang Zhang, Ting Pan, Mengyu Jia, Danyang Shi, and Wenjun Wang are with the College of Intelligence and Computing, Tianjin University, Tianjin 300350, China (e-mail: guoxuan@tju.edu.cn; wangzhang@tju.edu.cn; tingpan@tju.edu.cn; myjia@tju.edu.cn; shidanyang@tju.edu.cn; wjwang@tju.edu.cn).

Pengfei Jiao is with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China (e-mail: pjiao@hdu.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2023.3234005.

Digital Object Identifier 10.1109/TNNLS.2023.3234005

I. INTRODUCTION

NETWORK embedding (NE) [1], [2] has been a rolling network science and representation learning bandwagon in recent years. Researchers' enthusiasm for NE stems from its ability to transform large-scale unstructured data into low-dimensional structured representations. It brings convenient and efficient solutions to a great number of tasks, such as node classification [3], [4], link prediction [5], [6], and knowledge reasoning [7], [8].

The outbreak of NE research dates back to DeepWalk [9] that represents nodes in homogeneous networks. Hitherto, on homogeneous networks, almost all the methods have been devised to individually or simultaneously capture two complementary properties [10]: 1) proximity and 2) structural similarity. The methods on the former, for example, DeepWalk, aim to preserve the closeness among nodes into embeddings, while

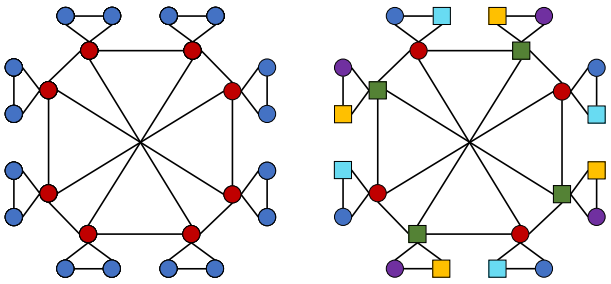

Fig. 1. Synthetic pinwheel networks. Node shapes denote their types and node colors denote their structural roles. (a) Homogeneous and (b) heterogeneous pinwheels.

the methods capturing the latter, for example, struc2vec [3], try to make embeddings discriminative on different structural patterns (or roles) [11]. For example, the red nodes and blue nodes having different structures in Fig. 1(a) are in two structural roles.

In recent years, structural role-based NE has attracted increasing attention, as it can be of great help in learning the functions and behaviors of nodes [12]. In essence, these methods usually rely on or imitate methods of structural feature extraction [13], [14] and subgraph isomorphism test [15], [16]. It is worth noting that a series of them are developed based on **anonymous walks (AWs)** [17], [18], [19] to generate structural embeddings or enhance the hot field of graph neural networks (GNNs). Because it is theoretically proved that neighborhood structures of a node can be reconstructed with AWs starting from it [20].

In the real world, heterogeneous networks are more common than homogeneous networks, because the corresponding nodes of entities in the real world usually have multiple types. The heterogeneous structures are much more complex because the status can be amazingly diverse when different connection patterns meet various node types.[1] We give an example via Fig. 1 to demonstrate this fact. The heterogeneous pinwheel has the same connecting patterns as the homogeneous pinwheel. However, with just one more node type, the number of node roles in the former network is three times that of the latter. Therefore, learning representations on heterostructures is a more comprehensive issue, as representing homogeneous structures is only a special case of it.

Nevertheless, little attention has been paid to heterogeneous structural NE. Although NE on heterogeneous networks is also thriving [21], almost all existing heterogeneous NE methods can be considered as extensions of those proximity-based homogeneous works in the view of topology. For example, metapath2vec [22] and heterogeneous information network to vector (HIN2vec) [23] extend DeepWalk and explore meta-paths with biased random walks. Relational graph convolutional network (R-GCN) [24], HetSAGNN [25], and heterogeneous graph transformer (HGT) [26] are aware of relation types with GNN architectures. Thus, these methods flounder on learning heterostructures. To our best knowledge, node2bits [27] is the only existing method that tries to fuse both node types and structural features into embeddings. However, it does not consider the combination of node types and underlying structures as a whole (i.e., heterostructure). And its random walk-based feature aggregation and hashing style cannot learn delicate underlying structures.

To fill the big void of NE on heterostructures, we present a terrific amount of work in this article. The key point is to discriminate the structures mixed with various node types. To this end, we first propose a novel technology, **heterogeneous AW (HAW)**, where AW is a special case of it on homogeneous networks. And, we prove that one can reconstruct heterogeneous neighborhood heterostructures when the distribution of HAWs is known. However, the number of possible HAWs of a given length is usually excessive, and as a result estimating the exact distribution is practically impossible. To avert this problem, we provide **HAW embedding (HAWE)** that samples HAWs to capture heterostructures in a data-driven manner and learns node embeddings by predicting HAWs starting from each node. In addition, we design biased HAW (BHAW) and coarse HAW (CHAW) for more applicability, and the BHAWE and CHAWE that employ the motioned embedding mechanism on BHAWs and CHAWs, respectively. Finally, we conduct sufficient and intuitive experiments on synthetic and real-world networks[2] that we process for building the first benchmark on heterostructure learning. Compared with both classic homogeneous and heterogeneous NE methods, our HAWE and its variants perform outstandingly in representing heterostructures. In summary, our contributions are listed as follows.

1) As far as we know, we are the first to directly study NE that captures heterostructures.
2) We propose the novel HAW which has a theoretical guarantee on reconstructing heterogeneous neighborhood structures, and its more practical variants BHAW and CHAW.
3) We provide an effective heterogeneous structural NE method HAWE. It generates embeddings by predicting HAWs starting from each node. We also give its variants BHAWE and CHAWE that applies the same mechanism on BHAWs and CHAWs, respectively.
4) We give the first benchmark on heterostructure learning. The sufficient and intuitive experiments show that our methods achieve excellent performance in distinguishing heterostructures and can be applied to large-scale networks.

The rest of this article is organized as follows. Section II introduces the critical concepts and definitions. In Section III, we provide the details of our proposed method. Section IV verifies the effectiveness of our method compared with numerous baselines on sufficient experiments. Section V is about the related work. We conclude this article and give some ideas about our future works in Section VI.

II. Preliminaries

In this section, we formally give the problem definition and introduce the AW [20]. The main notations of this article are summarized in Nomenclature.

[1]For simplicity, we borrow the electronic term "heterostructures" below to refer to the diverse heterogeneous structures.

[2]Both the code of our methods and used data in this article can be found at github.com/naihemeng/HAWE.

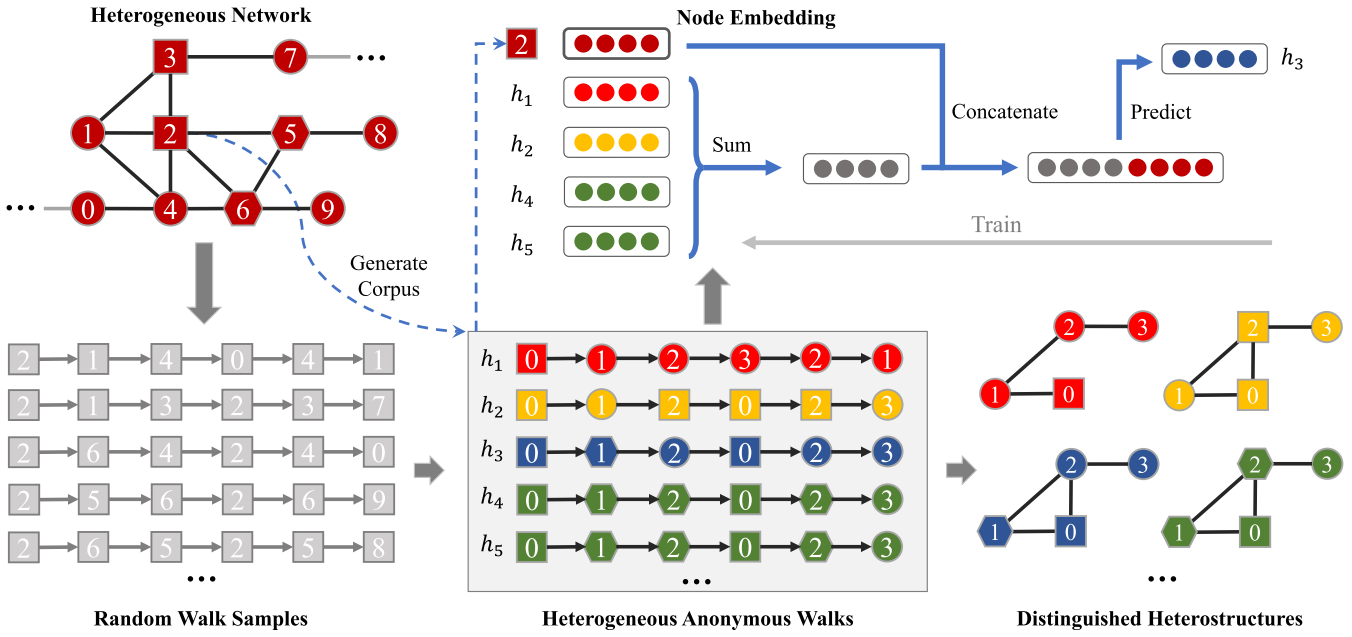

Fig. 2. Examples of HAW and the overview of proposed HAWE. The node shapes denote node types. The walks in the same color (h_4 and h_5) capture the same heterostructure.

A. Problem Definition

Definition 1 (Homogeneous Network): A homogeneous network is represented as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges.

Definition 2 (Heterogeneous Network): A heterogeneous network is defined as $\mathcal{H} = (\mathcal{V}, \mathcal{E}, \Phi, \Psi)$. Here, nodes $\mathcal{V}$ and edges $\mathcal{E}$ construct the topology. Φ maps each node $v \in \mathcal{V}$ to a node type $\Phi(v) \in \mathcal{T}$, where $\mathcal{T}$ is the node type set. And Ψ maps each edge $e \in \mathcal{E}$ to an edge type $\Phi(e) \in \mathcal{T}_{\mathcal{E}}$.

Note that the type of an edge is determined by the node type of its two ends in most heterogeneous networks. We omit edge types in this article. And we do not consider the self-loops.

Definition 3 (Network Embedding): Given a homogeneous/heterogeneous network $\mathcal{G}/\mathcal{H}$, NE aims to map $\mathcal{V}$ to low-dimensional vector representations $\{\mathbf{z}_v \in \mathbb{R}^d | v \in \mathcal{V}\}$, where the embedding dimension d is usually much smaller than $|\mathcal{V}|$. The obtained embeddings $\{\mathbf{z}_v\}_{v\in\mathcal{V}}$ should capture as rich information that benefits downstream tasks as possible.

In this article, we do not learn different mappings for different types of nodes, as we consider node types are an important part of the heterostructures. To evaluate the ability to capture heterostructures of our and baseline methods, we consider conducting structural role classification experiments and constructing a targeted benchmark.

B. Anonymous Walks

Definition 4 (Random Walk): A random walk is a sequence of nodes $w = (v_0, v_1, \ldots, v_l)$ where each latter node v_{i+1} is sampled uniformly from the neighbor set of the previous one $\mathcal{N}(v_i)$, $0 < i \leq l$. The length of w is l.

Given a network, generating random walks is an effective and efficient way to sample subsets of nodes and edges. While learning the structural patterns formed by the sampled nodes and edges, the node Id information is redundant. Thus, an anonymization process on random walks is usually applied.

Definition 5 (Anonymous Walk): Given a random walk $w = (v_0, v_1, \ldots, v_l)$, the corresponding AW of it is the l-length sequence of integers $a = (f(v_0), f(v_1), \ldots, f(v_l))$. f is the position function over w defined as $f(v_i, w) = |\{v_0, \ldots, v_{i'}\}|$, where $\{v_0, \ldots, v_{i'}\}$ is a subsequence of w starting from v_0 and i' is the smallest integer j satisfying $v_j = v_i$. For simplicity, we omit w later when denoting the position of a node in it by f.

See the lower-left part of Fig. 2 for some AW examples, where the node types are ignored. After anonymization, the shown five different random walks are mapped to two AWs (0, 1, 2, 3, 2, 1) and (0, 1, 2, 0, 2, 3). Meanwhile, the two AWs can be regarded as two kinds of subgraphs: 1) fourth-path and 2) tailed-triangle, respectively. Therefore, with AWs, we can learn higher-order structures similar to graphlets that are considered the building blocks of networks and node functions [28].

Theorem 1: The number of all possible AWs of the given length l is B_l. B_l is the lth Bell number that can be computed recurrently: $B_l = \sum_{k=0}^{l-1} \binom{l-1}{k} B_k$, where B_k starts from $B_0 = B_1 = 1$.

Proof: Consider anonymizing the occurred edges instead of nodes in a given AW so that each AW can be translated to a unique new sequence. For example, the AW (0, 1, 2, 0, 1) now can be represented as (0, 1, 2, 0). Then, the counting possible AWs is equivalent to the problem of counting rhyme schemes [29], by which the theorem can be proved. □

Theorem 2 [20]: Let $\mathcal{G}_r(v)$ be a homogeneous subgraph of which all nodes are within r distance from node $v \in \mathcal{V}$. Then one can reconstruct $\mathcal{G}_r(v)$ with P_l, where P_l is the distribution of l-length AWs starting from v with $l = 2(m + 1)$, and m is the number of edges in $\mathcal{G}_r(v)$.

Theorem 2 guarantees that structures can be captured with AWs. However, according to Theorem 1, estimating the exact distribution of AWs is hard, even though the used AWs are not very long. Thus, sampling strategies are always used on sizeable scale networks in previous AW-based NE works [17], [18], [19].

III. Method

A. Heterogeneous AWs

As stated, we are interested in learning the heterostructure, a mixture of different structural patterns and node types. Inspired by AWs having excellent ability to capture structures, we consider proposing a technique that can capture underlying structures as AWs do and identify the node types at the same time. Thus, we give the following definition of HAW.

Definition 6 (Heterogeneous AW): Given a random walk $w = (v_0, v_1, \ldots, v_l)$ occurring in a heterogeneous network $\mathcal{H} = (\mathcal{V}, \mathcal{E}, \Phi)$, its corresponding HAW is the l-length sequence of 2-tuples $h = (g(v_0), g(v_1), \ldots, g(v_l))$, where the tuple $g(v_i) = (f(v_i), \Phi(v_i))$.

As shown in the bottom part of Fig. 2, the four random walks that mapped to the same AW (0, 1, 2, 0, 2, 3) now can be distinguished as three unique HAWs. Each of these HAWs can be used to construct a heterogeneous subgraph. Although the three constructed subgraphs share the same underlying structures, that is, tailed triangles, they are different because of the different distributions of node types. We can still consider the constructed subgraphs as a new kind of graphlets. Empirically, our HAWs are equivalent to position-aware typed graphlets [30] which are aware of not only the type, but also the positions of each node.

More theoretically, we show that HAWs can be used to reconstruct the neighborhood heterostructures of a given node by the following theorem.

Theorem 3: Let $\mathcal{H}_r(v)$ be a heterogeneous subgraph of which all nodes are within r distance from node $v \in \mathcal{V}$. One can reconstruct $\mathcal{H}_r(v)$ with Q_l, where Q_l is the distribution of l-length HAWs starting from v with $l = 2(m+1)$, and m is the number of edges in $\mathcal{H}_r(v)$.

Proof: Here, we just provide a nonconstructive proof since our aim is not to design a practical reconstruction algorithm.

We call a HAW h economical if any ordered pairs $((s, \Phi(f^{-1}(s))), (t, \Phi(f^{-1}(t))))$ occurs at most once in it. Define $\mathcal{S} = \bigcup_{i=1}^{l} supp(Q_i)$. Let $H(h) = (V(h), E(h), \Phi')$ be the heterogeneous network reconstructed from $h = (g(v_0), g(v_1), \ldots, g(v_l))$, where $V(h) = \{f(v_i) | 0 \le i \le l\}$, $E(h) = \{(f(v_i), f(v_{i+1})) | 0 \le i < l\}$, and Φ' satisfies $\Phi'(s) = \Phi(f^{-1}(s))$. Then we have $h \in \mathcal{S} \iff H(h)$ is isomorphic to a heterogeneous subgraph of $\mathcal{H}$ and the start node v_0 of h is mapped to $f(v_0) = 0$ in $H(h)$. Thus, we can enumerate over $\mathcal{S}$ to find the longest $h^* \in \mathcal{S}$ such that both h^* is economical and $H(h^*)$ is of radius r from its central node 0. It is intuitive that $H(h^*)$ is isomorphic to $\mathcal{H}_r(v_0)$ which is a subgraph of $\mathcal{H}$, too. The $h^* \in \mathcal{S}$ must exist as any of the longest economical h of length $2(m+1)$ covers at least $m+1$ edges of its $H(h)$. □

Though HAWs can be used to capture the heterostructure of each node in light of Theorem 3, the ideal situation for achieving this is that we know the exact distribution of HAWs. Therefore, we wonder how difficult it is to estimate the exact HAW distribution.

Theorem 4: The number of all possible HAWs of the given length l is $|\mathcal{T}|^l B_l$, where there are $|\mathcal{T}|$ node types in the given heterogeneous network.

According to Theorem 4, the number of all possible HAWs grows exponentially with respect to walk length and is much more rapid than that of AWs. Although $|\mathcal{T}|^l B_l$ is just the upper bound for a specific heterogeneous network, computing HAW distribution is almost impossible, since the distribution of node types is always complex. Thus, we choose to use a sampling strategy to more efficiently take advantage of HAWs.

B. HAW Embedding

In this part, we consider how to preserve the heterostructure information captured by each HAW sample of a node into the corresponding embedding. Unlike proximity-based NE leveraging edges or other closeness measures as the optimization goal [22], [24], [31], [32], the substructure-based relations among nodes are unclear and the goal is hard to design. To solve the problem, some previous works, such as AWE [33] and GraphSTONE [18], draw lessons from the natural language process. They consider that the word-based relations among paragraphs are similar to the substructure-based relations among nodes. We introduce this idea to our embedding method.

From this perspective, we treat the neighborhood of each node as a text paragraph, and the sampled HAWs starting from the same node as the context words that occur together in the corresponding paragraph. The neighborhood heterostructures of a node usually have a theme. For example, an active user in a question-and-answer network behaves like a star-center node because of his/her large amounts of activities, that is, proposing questions and answers, while an expert user may have much less activities but propose much more answers than questions. The heterostructure theme can be expressed by the context words, that is, sampled HAWs, and makes it possible to predict a word in the context when knowing some other context words. Thus, we can construct an embedding model by learning this predictability.

Based on the above idea, we propose HAWE, a method to learn node representations on heterostructures. HAWE imitates the way that the language model distributed memory model of paragraph vectors (PV-DM) [34] generates paragraph representations with word representations. Its overview illustration is shown in Fig. 2.

Like learning language models, we need to generate a heterostructure corpus first. For each paragraph node $v \in \mathcal{V}$ in a given heterogeneous network $\mathcal{H}$, we sample a sequence of T HAWs $C^v = (h_1^v, h_2^v, \ldots, h_T^v)$ starting from it as its context words. Let $\mathcal{L}$ be the HAW lexicon over the corpus $\{C^v\}_{v \in \mathcal{V}}$. The size of $\mathcal{L}$ is always smaller than that of the total corpus since there are many common HAWs over all the contexts. And for different nodes, it is their common HAWs that reflect the neighborhood heterostructure similarities between them.

On the corpus $\{C^v\}_{v\in\mathcal{V}}$, HAWE learns a set of node embeddings $\{\mathbf{z}_v \in \mathbb{R}^d\}_{v\in\mathcal{V}}$ as well as a set of HAWEs $\{\mathbf{w}_h \in \mathbb{R}^d\}_{h\in\mathcal{L}}$, where d is embedding dimension of both node embeddings and HAWEs. For simplicity, we denote the HAWEs corresponding to the HAWs in the sequence $C^v = (h_1^v, h_2^v, \ldots, h_T^v)$ as $\mathbf{w}_1^v, \mathbf{w}_2^v, \ldots, \mathbf{w}_T^v$. HAWE establishes the relations among the paragraphs (i.e., nodes) and words (i.e., HAWs) by predicting words based on the paragraphs they belong to and their contexts sampled via a sliding window of length Δ. More formally, we train HAWE by maximizing the average log probabilities for all the words occurring in the corpus $\{C^v\}_{v\in\mathcal{V}}$ as follows:

$$\frac{1}{|\mathcal{V}|}\frac{1}{T}\sum_{v\in\mathcal{V}}\sum_{t=\Delta}^{T-\Delta}\log p\left(\mathbf{w}_t^v \middle| \mathbf{w}_{t-\Delta}^v, \ldots, \mathbf{w}_{t+\Delta}^v, \mathbf{z}_v\right) \tag{1}$$

where each probability is computed via the softmax function

$$p\left(\mathbf{w}_t^v \middle| \mathbf{w}_{t-\Delta}^v, \ldots, \mathbf{w}_{t+\Delta}^v, \mathbf{z}_v\right) = \frac{e^{y(\mathbf{w}_t^v)}}{\sum_{h\in\mathcal{L}} e^{y(\mathbf{w}_h)}}. \tag{2}$$

The un-normalized prediction probability $y(\mathbf{w}_t^v)$ for word h_t^v is computed as follows:

$$\widehat{\mathbf{w}}_t^v = \mathrm{Sum}\left(\mathbf{w}_{t-\Delta}^v, \ldots, \mathbf{w}_{t+\Delta}^v\right) \tag{3}$$

$$y\left(\mathbf{w}_t^v\right) = b + \mathbf{u}^\top\left[\widehat{\mathbf{w}}_t^v, \mathbf{z}_v\right] \tag{4}$$

where $b \in \mathbb{R}$ and $\mathbf{u} \in \mathbb{R}^{2d}$ are learnable parameters. Here, we sum over the context embeddings together to preserve as much heterostructure information as possible. Because we consider (3) as the pooling mechanism that GNNs need to apply after massage-passing. And it is proven that sum-pooling is more effective that other simple poolings such as mean- and max-pooling [13], [35].

In light of the above designs, the node embedding $\mathbf{z}_v$ is shared only across the contexts in which all the HAWs start from node v and has nothing to do with other contexts. Therefore, the different heterostructures can be differentiated by the node embeddings $\{\mathbf{z}_v\}_{v\in\mathcal{V}}$ trained via maximizing (1). And the nodes having similar HAW contexts would have similar embeddings. However, computing the denominator part of (1) needs a very high cost in practice. To mitigate this problem, we use hierarchical softmax [36] to speed up the computation by replacing the multiclass classification task with multilayers of binary classification tasks.

C. BHAW and HAWE

In practice, we can only sample a limited number of HAWs, so some key heterostructures could be missed. For example, the bridge edges should be more important than nonbridge edges for the involved nodes but are more difficult to be sampled due to their low frequencies. For more flexibility, we draw lessons from node2vec [37] and use two positive parameters p and q to bias HAWs. In a random walk having just traversed the edge (v^*, v), the unnormalized transition probability from node v to one of its neighbors $v' \in \mathcal{N}(v)$ in the next step is defined as

$$P\left(v'|v, (v^*, v)\right) = \begin{cases} 0, & \text{if } (v, v') \notin \mathcal{E} \\ 1/p, & \text{if } v' = v^* \wedge (v, v') \in \mathcal{E} \\ 1, & \text{if } (v', v^*) \in \mathcal{E} \wedge (v, v') \in \mathcal{E} \\ 1/q, & \text{if } (v', v^*) \notin \mathcal{E} \wedge (v, v') \in \mathcal{E} \end{cases} \tag{5}$$

where p and q control the HAWs to sample broader or deeper. In the later article, when we need to discriminate the biased method from the normal method, HAW and HAWE with $p \neq 1$ or $q \neq 1$ are especially called BHAW and BHAWE, respectively. With a proper setting of these two parameters, the BHAW can capture more important heterostructures while ignoring the inconsequential ones. And BHAWE might generate a smaller lexicon $\mathcal{L}$ than HAWE.

Algorithm 1 HAWE($\mathcal{H}$, Δ, d, T, L, p, q, Coarsening)

Input: the heterogeneous network $\mathcal{H} = (\mathcal{V}, \mathcal{E}, \Phi, \Psi)$; window size Δ; embedding dimension d; walk sample number per node T; walk length L; bias parameters p and q; Coarsening

Output: network embeddings $\{\mathbf{z}_v \in \mathbb{R}^d\}_{v\in\mathcal{V}}$

```
Initialize the corpus {C^v}_{v∈V} to Empty
for each v ∈ V do
    for i = 0 to T do
        Sample a HAW word h = HAW(H, L, p, q)
            ▷ Def. 6, Eq.(5): when p = q = 1, h is unbiased.
        if Coarsening then
            h = Coarsen(h)                                  ▷ Def. 7
        end if
        Append h to C^v
    end for
end for
Initialize embeddings {w_h ∈ R^d}_{h∈L} and {z_v ∈ R^d}_{v∈V}
PV-DM({z_v}_{v∈V}, {w_h}_{h∈L}, d, Δ)
    ▷ Eq.(1 − 4) : L is the lexicon over the corpus {C^v}_{v∈V}
return {z_v | v ∈ V}
```

D. CHAW and HAWE

In an ideal situation, the proposed HAWE can distinguish the heterostructure captured by each unique HAW. However, the design of HAW is not the most practical in many cases. On the one hand, the node embeddings capture similarities based on the common HAWs occurring in different contexts while ignoring the similarities between the heterostructures captured by different HAWs. On the other hand, on large-scale networks, the biggest HAW sample size for acceptable efficiency is still much smaller than the sample size required for a full understanding of the entire heterogeneous structure (Theorem 4). To deal with these problems, we propose a more practical variant of HAW.

Definition 7 (Coarse HAW): For an HAW $h = (g(v_0), g(v_1), \ldots, g(v_l))$ occurring in the given heterogeneous network $\mathcal{H} = (\mathcal{V}, \mathcal{E}, \Phi)$, its corresponding CHAW is the two-tuple $c = (a, \mathrm{OC}(h))$ composed of the lth AW $a = (f(v_0), f(v_1), \ldots, f(v_l))$ and the ordered node type

TABLE I

STATISTICS OF REAL-WORLD HETEROGENEOUS NETWORKS. ABBREVIATIONS OF NODE TYPES: A FOR AIRPORT (IN AIR-TRAFFIC)/ ANSWER (IN STACK EXCHANGE NETWORKS), C FOR COUNTRY, Q FOR QUESTION, U FOR USER

Dataset	#nodes	#edges	#classes
Air-traffic	A: 3,373, C: 226 | total: 3,599	A-A: 19,150, A-C: 3,373 | total: 22,523	A: 2
SE-Anime	A: 1,413, Q: 832, U: 234 | total: 2,479	A-Q: 1,413, A-U: 1,413, Q-U: 816 | total: 3,642	A: 3, U: 3
SE-Beer	A: 2,373, Q: 1,051, U: 1,150 | total: 4,574	A-Q: 2,373, A-U: 2,373, Q-U: 1044 | total: 5,790	A: 4, U: 3
SE-CG	A: 3,210, Q: 2,686, U: 1,480 | (total: 7,376)	A-Q: 3,210, A-U: 3,210, Q-U: 2,653 | total: 9,073	A: 3, U: 4
SE-Chem	A: 1,663, Q: 932, U: 387 | (total: 2,982)	A-Q: 1,663, A-U: 1,663, Q-U: 903 | total: 4,229	A: 3, U: 4
SE-CSE	A: 3,952, Q: 959, U: 1,224 | total: 6,135	A-Q: 3,952, A-U: 3,952, Q-U: 933 | total: 8,837	A: 4, U: 3
SE-Engr	A: 15,006, Q: 10,086, U: 6,855 | total: 31,947	A-Q: 15,006, A-U: 15,006, Q-U: 9,940 | total: 39,952	A: 4, U: 4
SE-FIT	A: 17,151, Q: 8,802, U: 6,748 | total: 32,701	A-Q: 17,151, A-U: 17,151, Q-U: 8,459 | total: 42,761	A: 5, U: 3
SE-HWR	A: 2,867, Q: 2,373, U: 1,788 | total: 7,028	A-Q: 2,867, A-U: 2,867, Q-U: 2,351 | total: 8,085	A: 4, U: 3
SE-IOT	A: 2,233, Q: 1,471, U: 1,231 | total: 4,935	A-Q: 2,233, A-U: 2,233, Q-U: 1,465 | total: 5,931	A: 3, U: 3
SE-Latin	A: 6,653, Q: 4,361, U: 1,619 | total: 12,633	A-Q: 6,653, A-U: 6,653, Q-U: 4,296 | total: 17,602	A: 4, U: 3
SE-Movie	A: 2,027, Q: 1,212, U: 461 | total: 3,700	A-Q: 2,027, A-U: 2,027, Q-U: 1,186 | total: 5,240	A: 3, U: 3

count $\mathrm{OC}(h) = ((\phi_0, n_0), (\phi_1, n_1), \ldots, (\phi_O, n_O))$. ϕ_i is the ith earliest kind of node type occurring in h and n_i is its presence frequency. $O \leq |\mathcal{T}l|$ is the number of all kinds of node types in h.

CHAWs count the node types and still remain a little but important position information by the order of the count list. In the view of graphlets, our CHAWs represent more delicate heterostructures than typed graphlets [30] which only care about node type presence frequency but totally ignore their positions and less delicate heterostructures than position-aware typed graphlets.

Intuitively, lots of HAWs capturing similar heterostructures are grouped and represented by a single CHAW so that the number of all possible CHAWs is much smaller than that of all possible HAWs. Thus, we can alleviate the above problems by replacing the HAWs in HAWE with corresponding CHAWs and getting the variant of embedding model CHAWE.

The pseudocode of the proposed methods is provided in Algorithm 1.

E. Time Complexity

Sampling an L-length HAW costs time of $O(L)$. Thus, the time complexity of obtaining heterogeneous corpus is $O(LT|\mathcal{V}|)$, as there are T HAWs sampled for each node. It takes $O(dT\Delta|\mathcal{V}|\log|\mathcal{L}|)$ for HAWE to learn representations from the heterogeneous corpus. Compared with HAWE, BHAWE, and CHAWE cost similar time on sampling but usually get a smaller lexicon $\mathcal{L}$ leading to less time cost on embedding generation. Considering the sparsity of real-world networks and most heterogeneous networks having a simple schema graph with no self-loops (i.e., there is no node having neighbors with the same type), $|\mathcal{L}|$ is usually much smaller than $T|\mathcal{V}|$. Therefore, the overall complexity of our methods is almost linear to the number of nodes.

IV. EXPERIMENTS

A. Real-World Datasets for Benchmark

Before our work, processed datasets for heterostructure learning are scarce. We construct several heterogeneous networks based on real-world datasets for building a benchmark on heterostructure learning, where the node labels indicate their structural roles. See Table I for some detailed statistics.

We construct an air-traffic network based on the data collected by OpenFlights.[3] It has two types of nodes: 1) airports (A) and 2) countries (C). The two types of edges denote air routes between airports and in which countries the airports are situated, respectively. The original dataset misses some detailed information about many airports including their countries. We fill in the missed information manually by searching the corresponding International Air Transport Association (IATA) or International Civil Aviation Organization (ICAO) codes online. We divide the airports into two classes based on their availability of international flights.

The others are Stack Exchange (SE) Q&A networks on different topics.[4] For simplicity, we refer to each of them as a prefix "SE" with a suffix of its topic abbreviation in the following article. They all have three types of nodes: 1) users (U); 2) questions (Q); and 3) answers (A). The three types of edges represent the user giving a question/answer and the answer answering a question, respectively. For each network, we group users based on their reputation (a measurement of how many upvotes/downvotes gained by the user from his/her questions and answers) and answers based on their scores (the difference between its upvotes and downvotes) into balanced multiple classes, respectively. There is also information about upvote and downvote counts for each user, which we use for more detailed demonstration in the experiment of similarity search.

To better understand the labels in SE networks, we compute Spearman correlation coefficients between the user reputation/answer score and some other statistics. The results are shown in Table II. We find the user reputation is more correlated with the number of users' Q-type neighbors than that of A-type neighbors, as questions are usually easier to gain votes than answers. All answers have the same degree as

[3]https://openflights.org/data.html, accessed October 2021.

[4]https://archive.org/download/stackexchange, accessed October 2021. Abbreviations of SE topics: Anime for anime & manga, CG for computer graphics, Chem for chemistry, CSE for computer science educators, Engr for engineering, FIT for physical fitness, HWR for hardware recommendations, IoT for Internet of Things, Latin for Latin language, Movie for movies & TV.

TABLE II
SPEARMAN CORRELATION COEFFICIENTS (WITH p-VALUES) ON STACK EXCHANGE NETWORKS. THE STRONGER CORRELATION IS IN BOLD

Dataset	User reputation & #A-type neighbors	User reputation & #Q-type neighbors	Answer score & U-type neighbor reputation	Answer score & A-Q-U-path neighbor reputation
SE-Anime	0.09 (0.15)	**0.52** (0.00)	**0.13** (0.00)	-0.03 (0.26)
SE-Beer	0.13 (0.00)	**0.47** (0.00)	**0.35** (0.00)	0.15 (0.00)
SE-CG	0.20 (0.00)	**0.28** (0.00)	0.31 (0.00)	**0.36** (0.00)
SE-Chem	0.06 (0.20)	**0.46** (0.00)	**0.16** (0.00)	-0.11 (0.00)
SE-CSE	0.11 (0.00)	**0.24** (0.00)	**0.18** (0.00)	0.11 (0.00)
SE-Engr	0.09 (0.00)	**0.23** (0.00)	0.11 (0.00)	**0.29** (0.00)
SE-FIT	0.07 (0.00)	**0.42** (0.00)	**0.39** (0.00)	0.21 (0.00)
SE-HWR	0.17 (0.00)	**0.32** (0.00)	0.28 (0.00)	**0.37** (0.00)
SE-IOT	0.13 (0.00)	**0.21** (0.00)	0.23 (0.00)	**0.43** (0.00)
SE-Latin	0.09 (0.00)	**0.47** (0.00)	**0.24** (0.00)	0.11 (0.00)
SE-Movie	0.13 (0.00)	**0.38** (0.00)	0.02 (0.45)	**-0.11** (0.00)

they only have one U-type neighbor and one Q-type neighbor. Thus, we consider the reputation of its U-type neighbor and A-Q-U-path neighbor. This is an embodying of regular equivalence [12] and the results verify it in most networks. However, the coefficients between the answer score and the two statistics are not at the same level in each network. This indicates that networks in different topics have different patterns. Thus, this series of networks could be used for method evaluation on generation capability. Note that Table II only gives an abecedarian analysis and verify the labels are related to heterostructure information. For better performance, the NE methods need to capture more complicated heterostructures.

B. Baseline Methods

To comprehensively understand the essence of our methods, we compare them with both homogeneous and heterogeneous NE methods. The homogeneous NE methods are as follows.

1) **DeepWalk** [9] treats a network as a document and the nodes as words. It leverages random walks to extract the contexts of each node and applies a language model to generate embeddings.
2) **LINE** [38] learns embeddings by reconstructing the first-order and second-order proximities between nodes.
3) **RolX** [39] learns nonnegative role-based embeddings by factorizing the effective structural feature matrix generated by ReFeX [40].
4) **Struc2vec** [3] constructs a multilayer complete graph on the nodes of the original network based on computed pair-wise structural similarities. A similar mechanism used by DeepWalk is then applied.
5) **GraphWave** [41] leverages heat wavelet diffusion patterns and learns structural embeddings via empirical characteristic functions of the wavelet coefficient distributions.
6) **Role2vec** [42] first assigns roles to nodes based on higher-order features and then applies a random walk-based embedding method in which it replaces node Ids with roles.
7) **GraphSTONE** [18] leverages AWs to capture structural patterns and gives a graph latent Dirichlet allocation (LDA) model to capture the structural topics of each node. A two-view graph convolutional layer is designed to fuse both structural similarity and proximity into embeddings.

When we apply these homogeneous NE methods, we ignore the node types in the input heterogeneous networks. The heterogeneous NE methods are as follows.

1) **HIN2vec** [23] uses random walks to generate node sequences. It learns the relations between nodes by predicting the meta-paths occurring in the sequences and covering the corresponding nodes.
2) **TransE** [43] learns embeddings by translating each triplet (a typed edge and its two endpoints) into embedding distance calculation.
3) **Node2bits** [27] designs biased random walks to aggregate neighbors' structures of each node and uses a hashing method to generate embeddings.
4) **R-GCN** [24] uses multiple graph convolutional layers to adaptively learn the corresponding type of relations among nodes. It is trained by reconstructing relations.
5) **HDGI** [44] is an unsupervised method inheriting the basic architecture of HAN [4], an attentive GNN model passing massages based on meta-paths.
6) **MAGNN** [31] proposes a massage-passing architecture that prevents from losing the semantics of the intermediate nodes in meta-paths and combines the massages from different meta-paths. It is trained by reconstructing relations.
7) **R-HGNN** [32] empowers representations to distinguish relation types among nodes by learning from relation-specific graphs and integrates the learned representations across different relations. It is trained by reconstructing relations.
8) **HGT** [26] models the heterogeneity among nodes and edges with attention mechanisms and generates type-specific embeddings. For each pair of nodes, it tries to preserve the triplet of two node types and the edge type to train the embeddings.

Note that RolX, struc2vec, GraphWave, role2vec, GraphSTONE, and node2bits are designed for structure learning. In all experiments, the parameters of these baseline methods are finely tuned. For heterogeneous deep graph infomax (HDGI), we employ predefined meta-paths to build typed adjacency matrices. Specifically, we use {ABA, ABBA, BAB, BAAB} (A for circle and B for square) on heterogeneous pinwheel network, {AA, ACA} on the air-traffic network. On SE networks, we apply {UAU, UQU, UAQU} for user classification and {AUA, AQA, AQUA} for answer

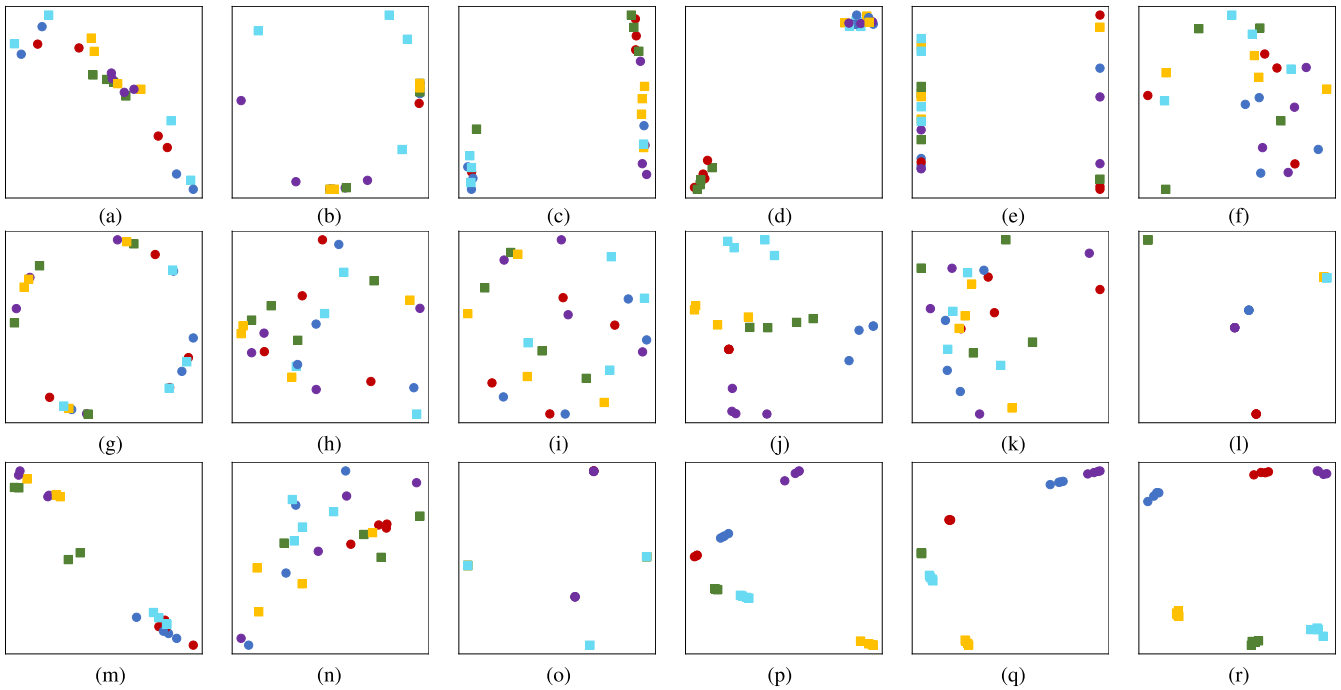

Fig. 3. Two-dimensional-embedding visualization for all the baselines and our methods on the synthetic heterogeneous pinwheel network. The point shapes denote node types and the point colors denote heterostructural patterns. (a) DeepWalk. (b) LINE. (c) RolX. (d) Struc2vec. (e) GraphWave + PCA. (f) Role2vec. (g) GraphSTONE. (h) HIN2vec. (i) TransE. (j) Node2bits + PCA. (k) R-GCN. (l) HDGI. (m) HGT. (n) MAGNN. (o) R-HGNN. (p) HAWE (Ours). (q) CHAWE (Ours). (r) BHAWE (Ours).

classification. The same sets of meta-paths are also used for metapath aggregated graph neural network (MAGNN) and relation-aware heterogeneous graph neural network (R-HGNN).

C. Model Configuration

Except in parameter sensitivity analysis, we do the following configuration for HAWE and its variants. We set sample size $T = 1024$ and window size $\Delta = 5$ on all the networks. For BHAWE, we set $p = q = 4^{-6}$, as long-path sampling from the root node is more needed in the involved networks. For user classification on SE networks, the walk length L is set to 4, while in other situations it is set to 6. Embeddings are trained via stochastic gradient descent for 100 epochs.

D. 2-D-Visualization

We employ our methods and most baseline methods on the synthetic heterogeneous pinwheel network shown in Fig. 1(b) and generate 2-D node embeddings. For GraphWave whose embedding dimension cannot be changed and node2bits whose hashing process is invalid with too small embedding dimension, we generate higher-dimensional embeddings (100-D for GraphWave and 64-D for node2bits) and transform them into 2-D space via principal component analysis (PCA). The 2-D visualization results are shown in Fig. 3.

The nature of some methods is intuitively shown: 1) DeepWalk; 2) LINE; 3) HIN2vec; 4) TransE; and 5) HGT make embeddings of neighbors close while RolX, struc2vec, and node2bits groups the nodes having the same neighborhood structures. We can observe that almost all the baseline methods cannot distinguish heterostructures. HDGI can gather all nodes of the same role at one point because the role can be distinguished via the predefined meta-paths. For example, the blue nodes have no neighbors through the meta-path ABBA. But HDGI does not truly capture the heterostructures as it does not show the relation between nodes having the same underlying structures (e.g., the red and purple green). As we argued, node2bits is the only baseline method that captures both the type and structure of the nodes that make up the heterostructures. However, its hashing process scatters the node in the same heterogeneous roles. All of our methods can effectively distinguish all six heterogeneous structural roles. And they do much better than node2bits as they make the nodes in the same role closer to lower-dimensional embeddings. And they more clearly show relations between the nodes having the same underlying patterns through their relative positions. The two node types are vertically distributed while the three underlying patterns are horizontally distributed in the same order. Compared with the other two proposed methods, BHAWE makes the clusters more uniform in the embedding space, as the long-path BHAWs can better discriminate node roles. Thus, the (B/C)HAWs do capture heterostructures and the embedding model can preserve them into representations.

E. Heterogeneous Structural Role Classification

We conduct heterogeneous structural role classification experiments on real-world networks. Specifically, we apply all methods on these networks and generate 128-D embeddings (100-D for GraphWave). For each method, we take 70% of

TABLE III
AVERAGE ACCURACY OF USER CLASSIFICATION ON THE STACK EXCHANGE NETWORKS

Method	SE-Anime	SE-Beer	SE-CG	SE-Chem	SE-CSE	SE-Engr	SE-FIT	SE-HWR	SE-IOT	SE-Latin	SE-Movie
DeepWalk	0.3611	0.4031	0.2895	0.2557	0.3967	0.2573	0.4105	0.3691	0.3663	0.3458	0.3859
LINE	0.4772	0.4574	0.3427	0.3532	0.4593	0.3201	0.4418	0.4209	0.4348	0.4191	0.4408
RolX	0.4755	0.5563	0.3699	0.3825	0.4937	0.3354	0.5169	0.4426	0.3374	0.4581	0.4439
struc2vec	0.4958	0.5055	0.3459	0.3574	0.4798	0.3348	0.4838	0.4132	0.4133	0.4373	0.4646
GraphWave	0.3383	0.3273	0.3301	0.3151	0.3567	0.3913	0.3935	0.4109	0.2901	0.3608	0.3742
role2vec	0.4146	0.4015	0.3536	0.3119	0.3644	0.4107	0.4399	0.4389	0.3463	0.4085	0.4087
GraphSTONE	0.4383	0.4713	0.3581	0.3260	0.4572	0.3091	0.4694	0.4256	0.3124	0.4085	0.4099
HIN2vec	0.4865	0.5193	0.3521	0.3653	0.4770	0.3128	0.4976	0.4175	0.4428	0.4046	0.4519
TransE	0.4963	0.5083	0.3483	0.3516	0.4453	0.3184	0.4811	0.4062	0.4093	0.4388	0.4426
node2bits	0.4924	0.5561	0.3544	0.3474	0.4851	0.3102	0.5112	0.4419	0.4374	0.4450	0.4905
R-GCN	0.3608	0.3346	0.2630	0.2646	0.3639	0.2574	0.3523	0.3426	0.3597	0.3579	0.3412
HDGI	0.4935	0.5489	0.3573	0.3651	0.4775	0.3921	0.4927	0.4128	0.4243	0.4279	0.4609
MAGNN	0.4961	0.5406	0.3647	0.3498	0.4544	0.2827	0.4689	0.4332	0.4288	0.4125	0.4584
R-HGNN	0.3783	0.3620	0.2715	0.2721	0.3435	0.2836	0.3742	0.3644	0.3646	0.3659	0.3562
HGT	0.4814	0.5242	0.3522	0.3473	0.4637	0.3094	0.4725	0.4259	0.4179	0.4391	0.4517
HAWE (ours)	**0.5307**	**0.5954***	**0.4166**	0.3615	0.4947	0.4152	**0.5673**	**0.5033**	**0.5138**	0.5147	**0.5028**
CHAWE (ours)	0.5177	**0.5927**	0.4069	**0.4231**	**0.5333***	**0.4273***	**0.5705***	0.4927	**0.5270***	**0.5292***	0.4868
BHAWE (ours)	**0.5501***	0.5863	**0.4222***	**0.4418***	**0.5194**	**0.4242**	0.5666	**0.5065***	0.5086	**0.5280**	**0.5374***

TABLE IV
AVERAGE ACCURACY OF ANSWER CLASSIFICATION ON THE STACK EXCHANGE NETWORKS

Method	SE-Anime	SE-Beer	SE-CG	SE-Chem	SE-CSE	SE-Engr	SE-FIT	SE-HWR	SE-IOT	SE-Latin	SE-Movie
DeepWalk	0.3781	0.3048	0.4617	0.4102	0.3089	0.3385	0.2838	0.3712	0.5033	0.3674	0.4025
LINE	0.3348	0.2581	0.3624	0.3358	0.2485	0.2573	0.2080	0.2581	0.3306	0.2531	0.3516
RolX	0.3753	0.3251	0.4175	0.4196	0.3083	0.3154	0.2889	0.3343	0.4457	0.3371	0.3924
struc2vec	0.3715	0.2905	0.3701	0.3732	0.2835	0.2911	0.2333	0.3078	0.3918	0.2819	0.3548
GraphWave	0.3056	0.2222	0.3627	0.3043	0.2347	0.3252	**0.3468**	0.2721	0.3887	0.3095	0.3533
role2vec	0.3619	0.3122	0.4727	0.3451	0.2832	0.3541	**0.3678***	0.4039	0.5428	0.3664	0.3538
GraphSTONE	0.3846	0.3043	0.4109	0.4054	0.2808	0.3073	0.2494	0.3792	0.4561	0.3439	0.4007
HIN2vec	0.3775	0.3202	0.4528	0.4298	0.3223	0.3455	0.3075	0.3794	0.5140	0.3664	0.4093
TransE	0.3672	0.3098	0.4022	0.4177	0.3111	0.2883	0.2613	0.3067	0.4483	0.3426	0.3802
node2bits	0.3963	0.3294	0.4213	0.4281	0.3172	0.3240	0.2914	0.3387	0.4414	0.3252	0.3745
R-GCN	0.3445	0.2538	0.3458	0.3489	0.2649	0.2494	0.2227	0.2585	0.3974	0.3103	0.3346
HDGI	0.2978	0.3298	0.3254	0.3307	0.2461	0.2147	0.2398	0.2718	0.3623	0.2586	0.3757
MAGNN	0.3332	0.2560	0.3301	0.3523	0.2505	0.2807	0.2248	0.3144	0.3318	0.2212	0.3413
R-HGNN	0.3301	0.2612	0.3432	0.3588	0.2724	0.2647	0.2204	0.3575	0.3426	0.2445	0.3544
HGT	0.3818	0.3165	0.3923	0.4259	0.2890	0.2897	0.2373	0.3056	0.4194	0.3181	0.3609
HAWE (ours)	**0.4106**	**0.3788**	0.5143	0.4336	0.3264	**0.3823***	0.3370	**0.4707***	**0.6249***	**0.3863**	**0.4159**
CHAWE (ours)	**0.4194***	**0.3805***	**0.5250***	**0.4468***	**0.3505***	0.3770	0.3342	0.4562	**0.6233**	0.3844	0.4131
BHAWE (ours)	0.4010	0.3693	**0.5184**	**0.4413**	**0.3458**	**0.3798**	0.3314	**0.4612**	0.6129	**0.3896***	**0.4324***

generated embeddings as the training set to train a Logistic Regression classifier. Then we apply the trained classifiers on the test set, that is, the other 30% embeddings, and calculate the classification accuracy. We repeat the above process 50 times and report average accuracy in Tables III–V. On each task, the top two results are bold while the best one is marked with the symbol *.

In the air-traffic (original) network, every domestic airport and all of its neighbor airports are connected to the same country node, while the neighbor airports of international airports may belong to different countries. Thus, the essential task is to detect a four-path subgraph in which two connected airports are connected to different countries, respectively (i.e., C-A-A-C). As designed for capturing this kind of heterostructure, HAWE and BHAWE unsurprisingly perform the best. As we bias the BHAWs to sample long paths, the corpus is more related to the task so that BHAWE achieves amazing results (see the detailed analysis in Section IV-G). And it is ineluctable for CHAWE to get lower results than HAWE because of the need for delicate detection. We can observe from the reported results that most methods achieve great performance. This is because domestic airports gather as communities so that methods of capturing proximities can also detect them. And international airports usually have a higher degree than domestic airports, which makes methods of learning structures work. For verification, we modify the original air-traffic network by deleting 50% of edges among the domestic airports and redoing the experiments on it. As expected, baseline methods capturing proximities get worse performance in the modified airport network, and the performance of those capturing structures almost stays unchanged. Accuracy of HAWE and its variants increases because the heterostructural traits of the two kinds of airports become more prominent.

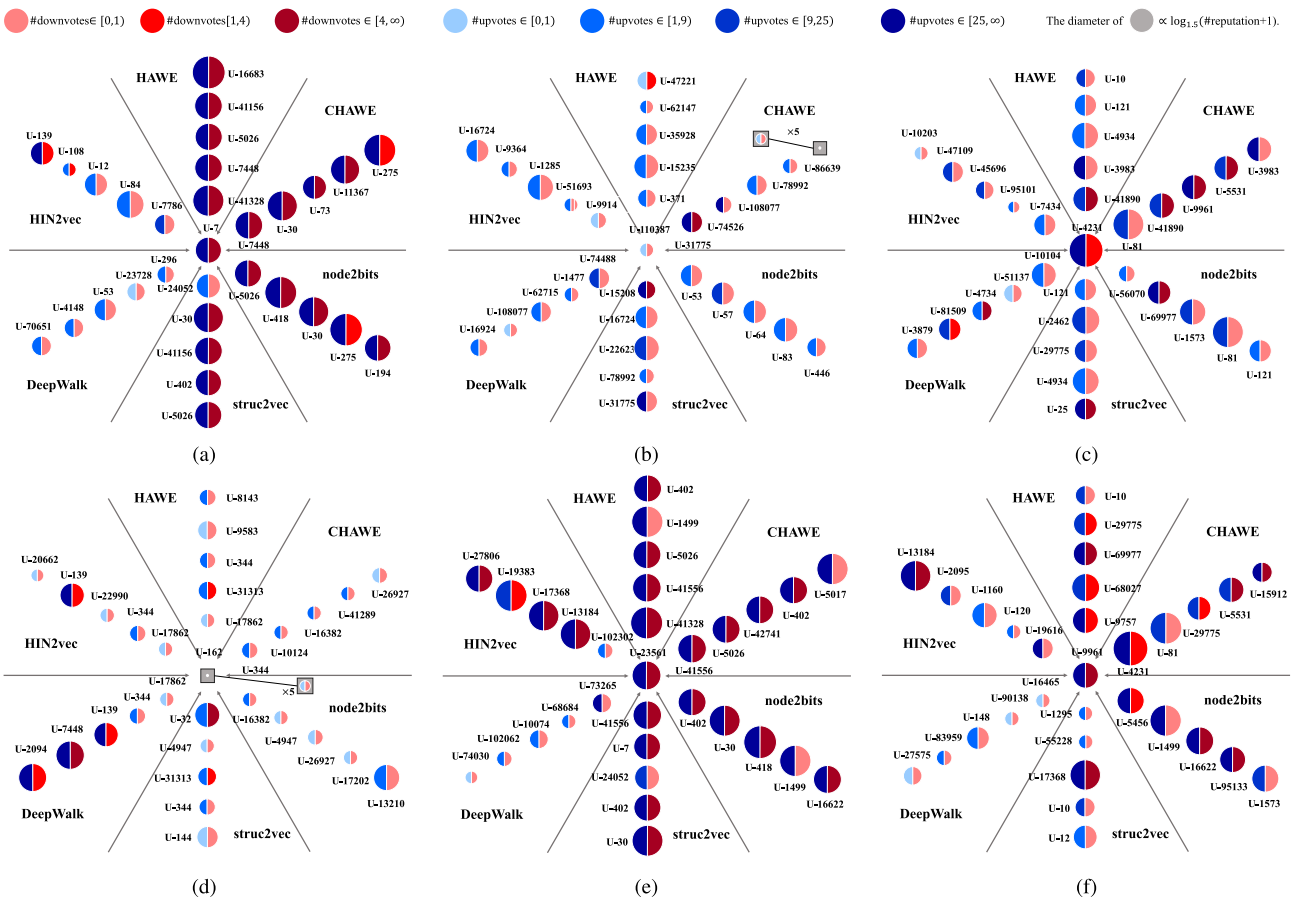

Fig. 4. Characteristic visualization of the closest five users in embedding space for some specific users. The circle size denotes the user's reputation and color depth denotes the upvote/downvote count of the user. (a) Earliest and (b) latest user. User with the (c) highest reputation, (d) lowest reputation, (e) most upvotes, and (f) most downvotes.

TABLE V
AVERAGE ACCURACY OF AIRPORT CLASSIFICATION ON THE ORIGINAL AND MODIFIED AIR-TRAFFIC NETWORKS

Method	Air-traffic (Original)	Air-traffic (Modified)
DeepWalk	0.8450	0.8269
LINE	0.8116	0.8002
RolX	0.8716	0.8829
struc2vec	0.8372	0.8333
GraphWave	0.8145	0.8258
role2vec	0.5348	0.5462
GraphSTONE	0.8288	0.8273
HIN2vec	0.9129	0.9051
TransE	0.8499	0.8501
node2bits	0.8607	0.8601
R-GCN	0.5249	0.5227
HDGI	0.8365	0.8523
MAGNN	0.7211	0.6962
R-HGNN	0.5221	0.5065
HGT	0.8961	0.8841
HAWE (ours)	**0.9242**	**0.9351**
CHAWE (ours)	0.9001	0.9158
BHAWE (ours)	**0.9889***	**0.9923***

On each SE network, we classify users and answers, respectively. On user classification, RolX, struc2vec, and node2bits outperform other baseline methods over most networks. These methods leverage statistical features which are strongly correlated with user reputation such as node degrees. The other structural embedding methods including GraphWave, role2vec, and GraphSTONE do not achieve competitive results due to the failure of their structure capture mechanisms (e.g., wavelet coefficient distributions and AWs) on heterogeneous networks. In contrast, the feature-based methods show no superiority compared with the other baseline methods on answer classification. This is because the score of the answer is influenced by both the reputation of the respondent and the quality of the answer itself. The latter is not related to the neighborhood heterostructures of answers and cannot be discriminated by all the compared methods. What is more, high-quality answers are usually provided by users having a high reputation. Thus, there is a proximity between high-score answers, which leads to the competitive results of some proximity-based methods such as DeepWalk and HIN2vec. However, in almost all networks, our methods get the top results on both user classification and answer classification because of their superiority in capturing heterostructures.

F. Similarity Search

We further design a top-k similarity search experiment to illustrate more details. We compute the Euclidean distances

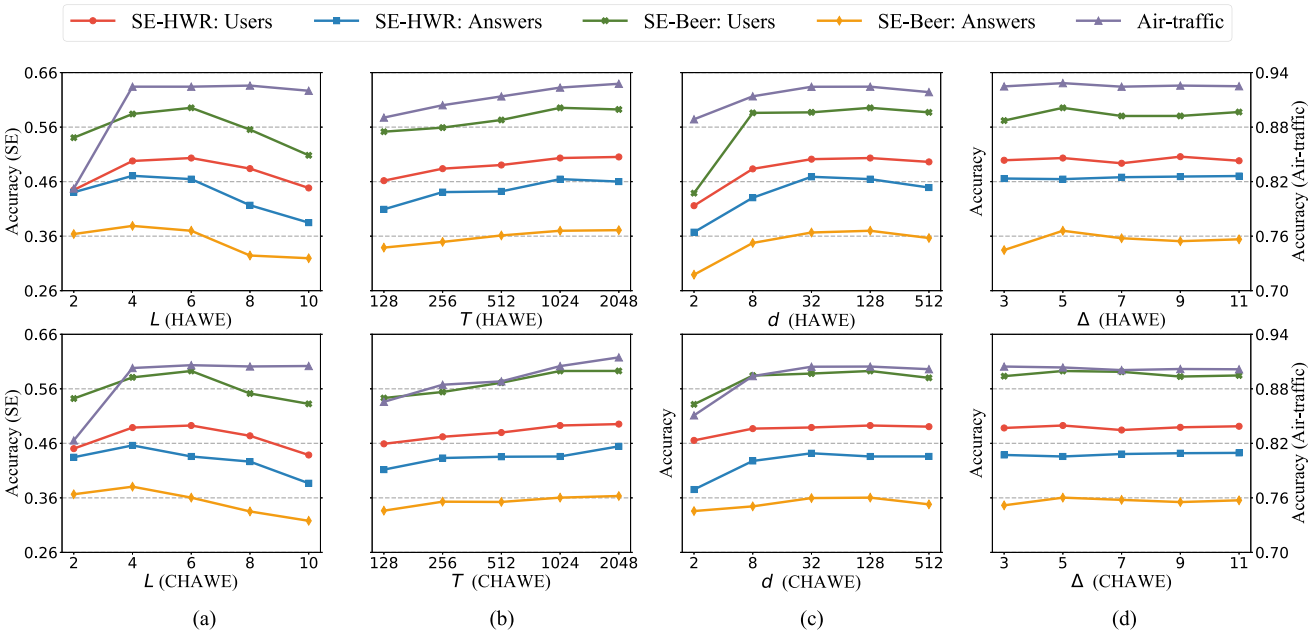

Fig. 5. Sensitivity analysis results on common parameters of our methods including: (a) walk length L, (b) sample number T, (c) embedding dimension d, and (d) window size Δ.

between user embeddings generated on the SE-Chem network by each method. Then we retrieve five users whose corresponding embeddings are the closest to that of a target user.

In Fig. 4, we show the search results of HAWE, CHAWE, node2bits, struc2vec, HIN2vec, and DeepWalk on six specific users. Each user is displayed as a circle consisting of two semicircles. The circle size denotes user reputation, and the semicircle color depth denotes the upvote/downvote count range of a user. The ranges are divided in a balanced manner. We also show the user IDs recorded in the raw data. The target users include: (a) the user (U-7) created the account the earliest in 2012; (b) the user (U-110387) created the account the latest in 2021; (c) the user (U-4231) having the highest reputation; (d) the user (U-162) having the lowest reputation; (e) the user (U-23561) having the most upvotes; (f) the user (U-9961) having the most downvotes. Their characteristics are provided in Table VI.

In the ideal case, the users retrieved by a method good at learning heterostructures should have similar circle sizes and semicircle color depths to the target user. With this criterion, we can observe that HAWE and CHAWE are in the top tier of performance. And CHAWE is better than HAWE due to the fuzzy design of CHAWs on similar heterostructures. Methods capturing structures, that is, node2bit and struc2vec achieve the level of the second tier. HIN2vec and DeepWalk often retrieve users with a relatively low reputation which verifies their nature of capturing proximities. Therefore, the results of the similarity search comprehensively show the superiority of our HAWE and CHAWE in learning heterostructures at the microlevel.

G. Parameter Sensitivity Analysis

In this part, we first study how the important parameters including walk length L, sample number T, embedding dimension d, and window size Δ influence the effectiveness of (B/C)HAWE. Specifically, we employ structural role classification on Air-traffic, SE-Beer, and SE-HWR networks with one parameter changing and the other parameters fixed. When the parameters are fixed, we set $L = 6$, $T = 1024$, $d = 128$, and $\Delta = 5$. The results are demonstrated in Fig. 5. We can observe that each parameter affects the two methods in the same way as follows.

1) When L is the variable, the accuracy increases first and then decreases with the growth of L. This is because the sampled (B/C)HAWs have trouble capturing heterostructures when they are too short. When the walk length is too long, the captured heterostructures are overly diverse, and much more samples are needed. The extreme point of L on answer classification is smaller than that on user classification. Because the roles (i.e., quality) of answers are more dependent on the roles of close nodes than the roles of users.
2) The accuracy increases with the growth of L. The more walks are sampled, the more delicate distributions of (B/C)HAWs are estimated and the models are trained with more training samples (1).
3) As embedding dimension d increases, the accuracy increases first for more heterostructure information preserved. Then it almost remains constant or decreases

TABLE VI
CHARACTERISTICS OF SOME USERS IN SE-CHEM

User ID	# reputation	# upvotes	# downvotes
U-7	5,868	26	9
U-110387	101	0	0
U-4231	79,505	132	2
U-162	1	0	0
U-23561	11,596	1,594	123
U-9961	4,202	297	297

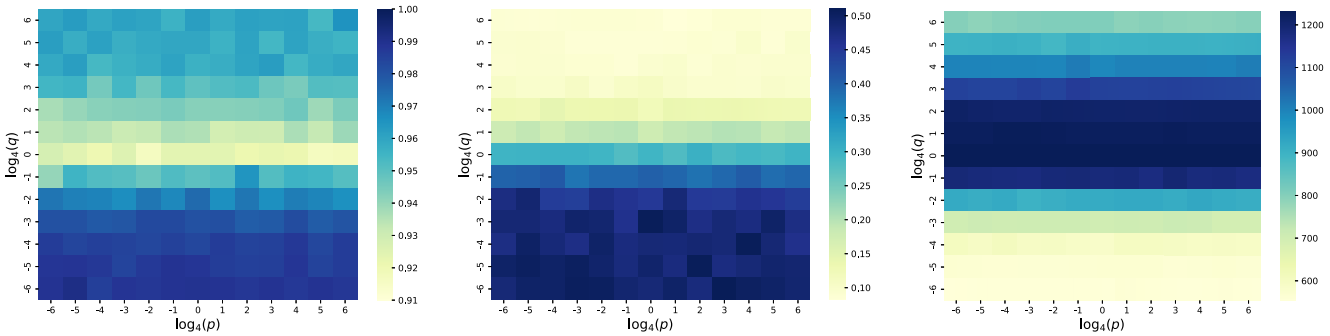

Fig. 6. Influence of walk-biasing parameters of BHAWE on the air-traffic network. (a) Accuracy. (b) Ratio of discriminative words. (c) Size of the lexicon.

slightly as the excessive embedding dimension is redundant for preserving heterostructure information and leads to an overfitting problem.

4) The window size Δ has little effect on the performance when it is large enough. When $\Delta \geq 5$, the information of (B/C)HAWs sampled by the sliding window is enough for predicting an unknown (B/C)HAW in the same context.

Then, we study how p and q affect BHAWE. We keep the other parameters fixed and try the combinations of p and q range between 4^{-6} and 4^6. We report the results on the Air-traffic network in Fig. 6. From Fig. 6(a), we find that p has almost no effects on the classification accuracy. But when $q > 1$ and $q < 1$, the accuracy is improved and the latter improves the results more significantly. To find out the reason for improvement, we study how the sampled corpus changed. We compute the ratio of the discriminative words for international airports, that is, the four-path C-A-A-C HAWs that the root node takes part in [see Fig. 6(b)]. The smaller q is, the ratio is larger due to the deeper sampling. Both $q < 1$ and $q > 1$ reduce the lexicon, but the former shows a more significant influence [see Fig. 6(c)]. Therefore, when $q \neq 1$, especially when $q < 1$, with the effect on sample discriminative words and lexicon size, BHAWE learns more task-relevant and discriminative representations.

H. Runtime Analysis

To evaluate the scalability of our proposed methods, we generate two series of synthetic graphs of which the node numbers range from 100 to 100 000 via Erdos-Renyi (ER) model [45] and Barabási-Albert (BA) model [46], respectively. For the ER model, we set the probability of linking two arbitrary nodes to $10/N$ so that the average node degree is approximately fixed to 5. For the BA model, we set the number of edges linked from a new node to existing nodes to 1 so that the generated graphs are trees. The nodes in these networks are aligned to two types randomly. We train HAWE and its variants on these synthetic graphs for 100 epochs with all the parameters fixed: $L = 6$, $T = 1024$, $d = 128$, and $\Delta = 5$. We run the methods on each network five times. The average runtime and lexicon size are illustrated in Fig. 7. Intuitively, the cost time of all methods is linear with the edge number of the network, which

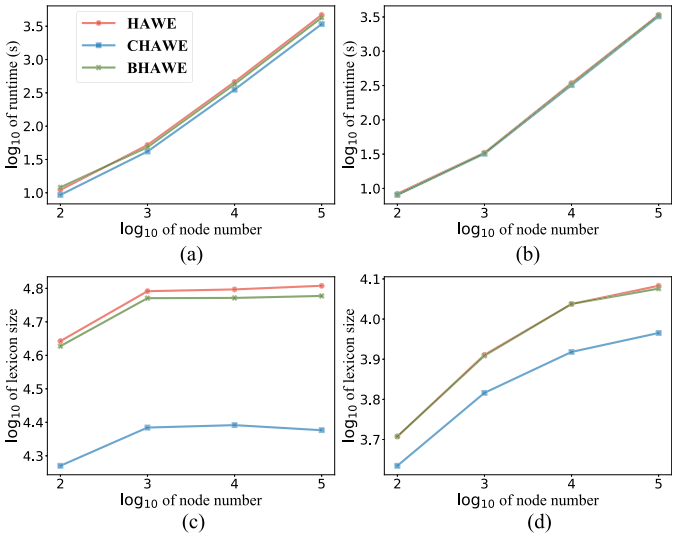

Fig. 7. Runtime and lexicon size of HAWE, CHAWE, and BHAWE on ER and BA graphs at a different scale. Runtime on (a) ER graphs and (b) BA graphs. Lexicon size on (c) ER graphs and (d) BA graphs.

verifies our conclusion derived in Section III-E. On ER graphs, (B)HAWE costs more time than CHAWE. On BA graphs, the runtime of the two methods is almost the same. This is because the (B)HAWE generates a much larger lexicon $\mathcal{L}$ than CHAWE on ER graphs, while the generated lexicons of these methods on BA graphs have small and similar sizes due to the tree structure. In real-world networks, since there are lots of loops, CHAWE is also more efficient than (B)HAWE.

V. Related Work

A. Homogeneous Structural Embedding

Almost all the existing algorithms of structural node embedding are designed for homogeneous networks [11], [47]. Usually, these works use a strategy of extracting structural properties such as extracting features [39], estimating wavelet distribution [41] and applying graph kernel method [16] before mapping them to vector space [11]. To transform captured structural traits into embeddings, early studies such as RolX [39] and guided learning for role discovery (GLRD) [48] directly factorize processed structural feature matrix. Later, a few works extend random walk-based methods (e.g., DeepWalk [9]). Struc2vec [3] achieves structural similar nodes

tending to occur in the same walk by constructing a new random walk graph based on node degrees. RiWalk [15] makes structural similar nodes have similar walks starting from themselves by proposing a new node labeling method. Recently, using deep learning to generate structural embeddings has attracted more attention. Deep recursive NE (DRNE) [49] captures regular equivalence by aggregating degrees of node neighbors with long short-term memory [50]. Curvature-based network embedding with stochastic equivalence (CNESE) [51] applies variational autoencoder [52] to learn stochastic equivalence by reconstructing the distributions of discrete graph curvature. Some works also introduce more complicated structural properties to guide the learning process [53], [54]. Graph neural networks with local structural patterns (GraLSP) [17] and GraphSTONE [18] use AWs to capture neighborhood structures and learn embeddings through GNN architecture.

Generally, though many structural NE methods have similar overall processes to proximity-based methods, they are very different in detailed steps [10], [11]. In the step of information encoding, structural NE usually leverages or approximates structural feature extraction [39], [51], graph kernels [15], or other approaches to capture the local or global structural patterns; proximity-based methods usually capture the closeness among nodes by random walks [9], message-passing mechanisms [55], and so on. In the step of training, the optimization goal is also specially designed to keep the captured properties. For structural NE, feature reconstruction [51] and structural similarity-based contrastive loss [3] are often used. For proximity-based NE, link reconstruction [18] and co-occurrence prediction [9] are common ideas. Many other choices can be used based on the characteristics of the datasets and tasks.

B. Heterogeneous NE

NE methods on a heterogeneous network are mainly devised for capturing the node proximity [21]. The earliest works of heterogeneous NE study knowledge graphs. These methods, such as TransE [43], train embeddings by learning a scoring function measuring how accurate embeddings represent the heterogeneous triplets. Later, a number of methods extended previous proximity-preserving homogeneous NE methods. Metapath2vec [22] and HIN2vec [23] extend DeepWalk [9] and preserve node proximity with random walks based on meta-paths. Predictive text embedding (PTE) [56] and AspEm [57] extend LINE [38] by designing edge-type-based and aspect-based proximities, respectively. Recently, heterogeneous GNNs (HGNNs) have become the main branch of heterogeneous NE. These methods use the massage-passing mechanism and most of them are semisupervised. For example, heterogeneous GNN (HetGNN) [58] uses random walks and bi-directional long short-term memory (Bi-LSTM) to capture neighborhood information. HAN [4] uses two levels of attention mechanism to aggregate neighborhood features. Heterogeneous graph structural attention neural network (HetSANN) [25] designs a type-aware attention layer to aggregate the embeddings of different types of nodes located in different spaces. It trains the embeddings through multitask learning. Unsupervised HGNNs have similar encoding mechanisms but are mainly designed for relation/link prediction. R-GCN [24] applies multiple GCNs [55] to learn the edge heterogeneity. MAGNN [31] aggregates node features within the meta-paths. HDGI [44], an unsupervised version of HAN on which the contrastive architecture of deep graph infomax (DGI) [59], is grafted. It uses both meta-path-based neighbor-level and semantic-level attention mechanisms to aggregate deep heterogeneous proximity information. HGT [26] uses a series of attention mechanisms to reconstruct the typed node pair in each triplet. R-HGNN [32] constructs relation-specific graphs to learn relation-aware and relation-crossing information.

To our best knowledge, existing heterogeneous NE methods except node2bits [27] cannot capture the highly diverse heterostructures. Node2bits extracts structural features for each type of neighbor sampled by random walks and generates embeddings via hashing methods. Its design for efficiency leads to low fineness in capturing heterostructures.

VI. Conclusion and Future Works

Learning the complex heterostructures, that is, the combinations of the node types and underlying structures, is a critical but underappreciated problem in the field of heterogeneous NE. In this article, we make a first attempt at NE on heterostructure learning. We propose the promising HAW which has a theoretically guaranteed ability to distinguish heterostructures and their more practical variants BHAW and CHAW. We take advantage of (B/C)HAWs by sampling them as the context of each node's heterostructure theme and provide an embedding method HAWE and its variants BHAWE and CHAWE by imitating a language model. Finally, we provide the first benchmark for learning heterogeneous structural roles. A number of datasets and tasks are proposed. As expected, our methods show amazing performance on them.

Although the proposed methods have a great ability to capture heterostructures, they are not perfect. Theoretically, the methods should achieve better performance when sampling longer (B/C)HAWs. But sampling overlong samples is not feasible due to the extreme sparsity of the samples. What is more, many sampled (B/C)HAWs are different but corresponded to the same induced heterogeneous subgraph. In other words, there are many synonyms in the generated heterostructure contexts. It is the inevitable result of the sampling strategy using (B/C)HAWs. But if we can reduce the number of synonyms, the effectiveness of our methods will be improved. We leave these problems as our future works.

References

[1] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Network representation learning: A survey," *IEEE Trans. Big Data*, vol. 6, no. 1, pp. 3–28, Mar. 2020.

[2] M. Boguna, I. Bonamassa, M. De Domenico, S. Havlin, D. Krioukov, and M. A. Serrano, "Network geometry," *Nature Rev. Phys.*, vol. 3, no. 2, pp. 114–135, 2021.

[3] L. F. R. Ribeiro, P. H. P. Saverese, and D. R. Figueiredo, "*struc2vec*: Learning node representations from structural identity," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 385–394.

[4] X. Wang et al., "Heterogeneous graph attention network," in *Proc. World Wide Web Conf.*, May 2019, pp. 2022–2032.

[5] P. Jiao et al., "Temporal network embedding for link prediction via VAE joint attention mechanism," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 12, pp. 1–14, Jun. 2021.

[6] Z. Wang, Y. Lei, and W. Li, "Neighborhood attention networks with adversarial learning for link prediction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 8, pp. 3653–3663, Aug. 2021.

[7] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 2, pp. 494–514, Feb. 2022.

[8] W. Zhang et al., "Iteratively learning embeddings and rules for knowledge graph reasoning," in *Proc. World Wide Web Conf. (WWW)*, 2019, pp. 2366–2377.

[9] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2014, pp. 701–710.

[10] R. A. Rossi, D. Jin, S. Kim, N. K. Ahmed, D. Koutra, and J. B. Lee, "On proximity and structural role-based embeddings in networks: Misconceptions, techniques, and applications," *ACM Trans. Knowl. Discovery Data*, vol. 14, no. 5, pp. 1–37, Oct. 2020.

[11] P. Jiao, X. Guo, T. Pan, W. Zhang, Y. Pei, and L. Pan, "A survey on role-oriented network embedding," *IEEE Trans. Big Data*, vol. 8, no. 4, pp. 1–20, Aug. 2021.

[12] R. A. Rossi and N. K. Ahmed, "Role discovery in networks," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 4, pp. 1112–1131, Apr. 2015.

[13] X. Guo, W. Zhang, W. Wang, Y. Yu, Y. Wang, and P. Jiao, "Role-oriented graph auto-encoder guided by structural information," in *Proc. Int. Conf. Database Syst. Adv. Appl.* Cham, Switzerland: Springer, 2020, pp. 466–481.

[14] R. A. Rossi, N. K. Ahmed, E. Koh, S. Kim, A. Rao, and Y. Abbasi-Yadkori, "A structural graph representation learning framework," in *Proc. 13th Int. Conf. Web Search Data Mining*, Jan. 2020, pp. 483–491.

[15] X. Ma, G. Qin, Z. Qiu, M. Zheng, and Z. Wang, "RiWalk: Fast structural node embedding via role identification," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2019, pp. 478–487.

[16] G. Nikolentzos and M. Vazirgiannis, "Learning structural node representations using graph kernels," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 5, pp. 2045–2056, Oct. 2019.

[17] Y. Jin, G. Song, and C. Shi, "Gralsp: Graph neural networks with local structural patterns," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 4361–4368.

[18] Q. Long, Y. Jin, G. Song, Y. Li, and W. Lin, "Graph structural-topic neural network," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 1065–1073.

[19] Q. Long, Y. Jin, Y. Wu, and G. Song, "Theoretically improving graph neural networks via anonymous walk graph kernels," in *Proc. Web Conf.*, Apr. 2021, pp. 1204–1214.

[20] S. Micali and Z. A. Zhu, "Reconstructing Markov processes from independent and anonymous experiments," *Discrete Appl. Math.*, vol. 200, pp. 108–122, Feb. 2016.

[21] C. Yang, Y. Xiao, Y. Zhang, Y. Sun, and J. Han, "Heterogeneous network representation learning: A unified framework with survey and benchmark," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 10, pp. 4854–4873, Oct. 2022.

[22] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 135–144.

[23] T.-Y. Fu, W.-C. Lee, and Z. Lei, "HIN2Vec: Explore meta-paths in heterogeneous information networks for representation learning," in *Proc. ACM Conf. Inf. Knowl. Manage.*, Nov. 2017, pp. 1797–1806.

[24] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *Proc. Eur. Semantic Web Conf.* Cham, Switzerland: Springer, 2018, pp. 593–607.

[25] H. Hong, H. Guo, Y. Lin, X. Yang, Z. Li, and J. Ye, "An attention-based graph neural network for heterogeneous structural learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 4, 2020, pp. 4132–4139.

[26] Z. Hu, Y. Dong, K. Wang, and Y. Sun, "Heterogeneous graph transformer," in *Proc. Web Conf.*, Apr. 2020, pp. 2704–2710.

[27] D. Jin, M. Heimann, R. Rossi, and D. Koutra, "node2bits: Compact time-and attribute-aware node representations," in *Proc. ECML/PKDD Eur. Conf. Princ. Pract. Knowl. Discovery Databases*, 2019, pp. 1–22.

[28] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, "Network motifs: Simple building blocks of complex networks," *Science*, vol. 298, no. 5594, pp. 824–827, 2002.

[29] M. Gardner, "Bells-versatile numbers that can count partitions of a set, primes and even rhymes," *Sci. Amer.*, vol. 238, no. 5, pp. 24–30, 1978.

[30] R. A. Rossi et al., "Heterogeneous graphlets," *ACM Trans. Knowl. Discovery Data*, vol. 15, no. 1, pp. 1–43, Feb. 2021.

[31] X. Fu, J. Zhang, Z. Meng, and I. King, "MAGNN: Metapath aggregated graph neural network for heterogeneous graph embedding," in *Proc. Web Conf.*, Apr. 2020, pp. 2331–2341.

[32] L. Yu, L. Sun, B. Du, C. Liu, W. Lv, and H. Xiong, "Heterogeneous graph representation learning with relation awareness," *IEEE Trans. Knowl. Data Eng.*, early access, Mar. 17, 2022, doi: 10.1109/TKDE.2022.3160208.

[33] S. Ivanov and E. Burnaev, "Anonymous walk embeddings," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 2186–2195.

[34] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1188–1196.

[35] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–17.

[36] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[37] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 855–864.

[38] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. 24th Int. Conf. World Wide Web*, May 2015, pp. 1067–1077.

[39] K. Henderson et al., "RolX: Structural role extraction & mining in large graphs," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2012, pp. 1231–1239.

[40] K. Henderson et al., "It's who you know: Graph mining using recursive structural features," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2011, pp. 663–671.

[41] C. Donnat, M. Zitnik, D. Hallac, and J. Leskovec, "Learning structural node embeddings via diffusion wavelets," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1320–1329.

[42] N. K. Ahmed et al., "Role-based graph embeddings," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 5, pp. 2401–2415, May 2022.

[43] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, 2013, pp. 1–9.

[44] Y. Ren and B. Liu, "Heterogeneous deep graph infomax," in *Proc. Workshop Deep Learn. Graphs, Methodologies Appl. Co-Located 34th AAAI Conf. Artif. Intell.*, 2020, pp. 1–9.

[45] E. N. Gilbert, "Random graphs," *Ann. Math. Statist.*, vol. 30, no. 4, pp. 1141–1144, 1959.

[46] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[47] J. Jin, M. Heimann, D. Jin, and D. Koutra, "Toward understanding and evaluating structural node embeddings," *ACM Trans. Knowl. Discovery Data*, vol. 16, no. 3, pp. 1–32, Jun. 2022.

[48] S. Gilpin, T. Eliassi-Rad, and I. Davidson, "Guided learning for role discovery (GLRD): Framework, algorithms, and applications," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2013, pp. 113–121.

[49] K. Tu, P. Cui, X. Wang, P. S. Yu, and W. Zhu, "Deep recursive network embedding with regular equivalence," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 2357–2366.

[50] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[51] X. Guo, Q. Tian, W. Zhang, W. Wang, and P. Jiao, "Learning stochastic equivalence based on discrete Ricci curvature," in *Proc. 30th Int. Joint Conf. Artif. Intell.*, Aug. 2021, pp. 1456–1462.

[52] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Learn. Represent.*, 2014, pp. 1–14.

[53] W. Zhang, X. Guo, W. Wang, Q. Tian, L. Pan, and P. Jiao, "Role-based network embedding via structural features reconstruction with degree-regularized constraint," *Knowl.-Based Syst.*, vol. 218, Apr. 2021, Art. no. 106872.

[54] P. Jiao, Q. Tian, W. Zhang, X. Guo, D. Jin, and H. Wu, "Role discovery-guided network embedding based on autoencoder and attention mechanism," *IEEE Trans. Cybern.*, vol. 53, no. 1, pp. 365–378, Jan. 2023.

[55] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–14.

[56] J. Tang, M. Qu, and Q. Mei, "PTE: Predictive text embedding through large-scale heterogeneous text networks," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2015, pp. 1165–1174.

[57] Y. Shi, H. Gui, Q. Zhu, L. Kaplan, and J. Han, "Aspem: Embedding learning by aspects in heterogeneous information networks," in *Proc. SIAM Int. Conf. Data Mining*, 2018, pp. 144–152.
[58] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla, "Heterogeneous graph neural network," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2019, pp. 793–803.
[59] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–4.

Xuan Guo is currently pursuing the Ph.D. degree with the College of Intelligence and Computing, Tianjin University, Tianjin, China.

His current research interests include complex network analysis, role discovery, network representation learning, and network percolation model.

Pengfei Jiao (Member, IEEE) received the Ph.D. degree in computer science from Tianjin University, Tianjin, China, in 2018.

From 2018 to 2021, he was a Lecturer with the Center of Biosafety Research and Strategy, Tianjin University. He is currently a Professor with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China. His current research interests include complex network analysis and its applications.

Wang Zhang received the bachelor's degree from Tianjin University, Tianjin, China, in 2018, where he is currently pursuing the master's degree with the School of Computer Science and Technology.

His current research interests include complex network analysis and network embedding.

Ting Pan received the bachelor's degree from Xiamen University, Xiamen, China, in 2020. She is currently pursuing the master's degree with the School of Computer Science and Technology, Tianjin University.

Her current research interests include complex network analysis and role-based network representation learning.

Mengyu Jia received the bachelor's degree from Hangzhou Dianzi University, Hangzhou, China, in 2018. She is currently pursuing the master's degree with the College of Intelligence and Computing, Tianjin University, Tianjin, China.

Her research interests include complex network analysis and heterogeneous representation learning.

Danyang Shi received the bachelor's degree from Tianjin University, Tianjin, China, in 2020, where he is currently pursuing the master's degree with the College of Intelligence and Computing.

His current research interests include heterogeneous information network representation learning.

Wenjun Wang is currently a Professor at the College of Intelligence and Computing, Tianjin University, Tianjin, China. He has published more than 50 papers in main international journals and conferences. His research interests include computational social science, large-scale data mining, intelligence analysis, and multilayer complex network modeling.