G-HIN2Vec: Distributed heterogeneous graph representations for cardholder transactions

Farouk DAMOUN*§, Hamida SEBA*§, Jean HILGER* , Radu STATE*
University of Luxembourg* Univeristy of Lyon 1§
{farouk.damoun, jean.hilger, radu.state}@uni.lu, {hamida.seb}@univ-lyon1.fr

ABSTRACT

Graph related tasks, such as graph classification and clustering, have been substantially improved with the advent of graph neural networks (GNNs). However, existing graph embedding models focus on homogeneous graphs that ignore the heterogeneity of the graphs. Therefore, using homogeneous graph embedding models on heterogeneous graphs discards the rich semantics of graphs and achieves average performance, especially by utilizing unlabeled information. However, limited work has been done on whole heterogeneous graph embedding as a supervised task. In light of this, we investigate unsupervised distributed representations learning on heterogeneous graphs and propose a novel model named G-HIN2Vec, Graph-Level Heterogeneous Information Network to Vector. Inspired by recent advances of unsupervised learning in natural language processing, G-HIN2Vec utilizes negative sampling technique as an unlabeled approach and learns graph embedding matrix from different predefined meta-paths. We conduct a variety of experiments on three main graph downstream applications on different socio-demographic cardholder features, graph regression, graph clustering, and graph classification, such as gender classification, age, and income prediction, which shows superior performance of our proposed GNN model on real-world financial credit card data.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; *Unsupervised learning*; *Learning latent representations*;

KEYWORDS

Heterogeneous Graph Embedding, Deep Learning, Financial Data

ACM Reference Format:
Farouk DAMOUN*§, Hamida SEBA*§, Jean HILGER* , Radu STATE* . 2023. G-HIN2Vec: Distributed heterogeneous graph representations for cardholder transactions. In *The 38th ACM/SIGAPP Symposium on Applied Computing (SAC '23), March 27-March 31, 2023, Tallinn, Estonia.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3555776.3577740

SAC '23, March 27–March 31, 2023, Tallinn, Estonia

ACM ISBN 978-1-4503-9517-5/23/03…$15.00
https://doi.org/10.1145/3555776.3577740

1 INTRODUCTION

Different transaction types are executed between multiple parties in financial institutions corresponding to other financial assets such as loans, investments, and insurance. Therefore, banks are omnipresent financial institutions in our daily lives, and financial card products dominate the retail banking market. As a result of digitization, the use of cashless payments with plastic/virtual cards has increased substantially over the past decade, with a high base that leads to an exponential growth in the digital footprint of card transactions [1].

Analyzing customer transactions is vital for various banking applications, such as behavior modeling, product recommendation, and advertising strategies [4]. Therefore, to extend and enrich the initial customer data, recent work related to financial data [10, 17] exposes the usefulness of the graph data structure to preserve topological information hidden in tabular data. On the other hand, more complexity to handle graph data is added, where each graph component has its attributes and local topology. The task becomes challenging when processing the entire graph, such as graph classification, clustering, and regression, which requires graphs to be in a vectorial representation, to apply machine learning algorithms. Graph kernel algorithms [22] as handcrafted feature extraction to overcome this problem, recently extended to deep learning models to provide dense latent graph encoding using embedding algorithms [8].

However, most embedding techniques are designed for homogeneous networks [8], where the graph schema has one node and edge type. Homogeneous networks are well studied and considered essential in analyzing graph data involving downstream machine leaning tasks over nodes, edges, and graphs [15, 29].

However, graph analysis tasks are limited by the problem setting and the embedding technique perspectives [3]. In node-level cases such as node regression and classification, the embeddings represent low-dimensional node representations, e.g., scientific publications classification using the citation network as node-level task. In the graph-level case, such as graph classification and regression tasks, the embedding represents a low-dimensional graph representation, e.g., protein function prediction using chemical compounds graphs, and malware detection using call graphs as graph-level task [11, 13].

However, in real-world scenarios, networks are more ubiquitous and consist of different types of interactions between graph components to provide an effective graph model, called heterogeneous information networks (HIN). Heterogeneous networks generalize the

[1]According to the Federal Prof. of Credit Card Operations of Depository Institutions report, 2020. ref.https://www.federalreserve.gov/publications/files/ccprofit2020.pdf

basic assumptions of homogeneous networks with different nodes and edges. Hence, graph neural networks (GNNs) are proposed as a class of deep learning-based methods for graphs with auxiliary information. GNNs are known for their robustness and effectiveness in different downstream graph-related tasks, but fail to fully preserve the network structure in both HG and HIN[25, 27]. However, the network structure is crucial in graph analytics, especially for HINs. Recently, random walk-based embedding methods [8] show considerable performance gains in different HG downstream tasks by preserving the network structure; but limited in heterogeneous schema. Meanwhile, most of the research work on graph embedding predominantly focuses on learning node representations.

Present work. Inspired by join learning architectures [20] and recent random walk-based methods for node embedding, we propose and develop a neural network architecture for a heterogeneous whole graph embedding named G-HIN2Vec, experimented with downstream retail banking tasks, as a data-driven KYC (know-your-customer). We list our main contributions as follows:

- We propose new graph modeling, the ego-centric graph model for cardholder transactions motivated by the egocentric thinking; to build an unlabeled heterogeneous graph dataset.
- We propose G-HIN2Vec, an unsupervised representation learning method for entire heterogeneous graph representations using double triplet loss function as task-agnostic approach.
- We experimentally benchmark different GNN models on real-world credit card datasets for different downstream tasks: graph classification, regression, and clustering tasks, where we achieve comparable results with G-HIN2Vec against the state-of-the-art models.

2 RELATED WORK

This section will review three research lines related to our work, namely heterogeneous information networks, whole graph embedding, and financial card transaction data analysis.

- **HIN Embedding:** Recently, Heterogeneous Information Network Embedding has drawn research attention to the use of complex graph modeling schemas with respect to real-world data that involve interactions of multitype objects. However, most of the HIN embedding algorithms used are developed in a limited context and tend to solve downstream tasks at the node or edge level [5, 7, 26], this is due to the lack of a unified framework and benchmark dataset and a baseline for HIN embeddings where the state of the art cannot be easily highlighted compared to HG [8]. Recent work [30] have experimented different HIN models in unified and controlled settings and benchmarked 13 state of the art algorithms in four datasets designed for node and edge level tasks; this work reveals that the nature of the task to be solved including the structural information and semantics of different entities in the HIN does not lead to one state of the art algorithm, as they were developed to solve a task-specific problem that depends on specific graph schema and other experimental factors, as also shown in [25, 27].
- **Whole-Graph Embedding:** In the literature, entire graph embedding algorithms are less studied than node level, edge level or subgraph embedding algorithms [13]. However, most graph-level embedding algorithms are used in graph-related tasks such as graph classification, regression, and clustering [14] with the same objective of preserving similarity in a graph dataset. As discussed in [13], HINs are transformed to HG dataset to use homogeneous graph embedding algorithms to generate graph-level embeddings, such as as graph2vec [14], and DGK [29], which shows improvement in regard to past performance. Others rely on existing heuristics for graph similarity, like UGraphEmb[1], this model trained on true distance matrix target labels in the form of a graph-graph pairwise distance metric, this tends to an expensive and not scalable approach to learn graph embedding. Unlike HG graphs, the HIN representation is generated by aggregate functions, such as avg layer, to transform the node into graph embeddings, this walk-around leads to suboptimal results [28].
- **Financial Card Transactions:** Research related to card transaction downstream tasks is predominated by the fraud detection use case, regarding the nature of the problem, in most cases it is considered a supervised problem. Most graph-based methods focus on simple graphs [17], even if the problem requires a multigraph schema to classify fraud and non-fraud transactions. This explains the recent interest in HIN-based GNNs for node embedding for different downstream tasks, such as in [11], HINs are used for the detection of malicious Alipay accounts, and in [16] for the detection of risky transactions in a B2B network, and in [32] for bankruptcy prediction in credit risk assessment. Besides fraud detection, card transaction data have drawn research attention to better understand transaction entities, for example, the use of node embeddings to better understand cardholder-merchant interactions [10], and for incentive optimization marketing [12], such works open the door for a variety of new use cases where HIN networks are essential to handle the heterogeneity of raw data, as mentioned in [24].

3 PROBLEM DEFINITION

Throughout this section, we introduce the notions of HIN, random metapath walks, and discuss the problem addressed in this work.

Definition 3.1. **Heterogeneous Information Network,** denoted HIN is defined as a network $G = (V, E, T_V, T_E, W)$ where G is a weighted directed graph, where V and E denote the set of nodes and edges, $W \in \mathbb{R}^{|V|x|V|}$ is the weight matrix, and T_V and T_E are type mapping functions for nodes and edges, respectively.

In HINs, node types are represented by $T_V : V \rightarrow N$, where N is a set of node types, and the relations between nodes are indicated by $T_E : V \times V \rightarrow R$, where R is a set of edge types.

For HINs, metapaths are used to generate random sequences guided by a predefined schema, defined as follows

Definition 3.2. **Random Meta-Path Walk.** Given the schema of G, a metapath $\mathcal{P}$ is defined as the sequence of triplets (n_i, r, n_j) where n_i *and* $n_j \in N$ are the source and target node types, and $r \in R$

is the edge type, $\mathcal{P}$ is denoted as follows

$$\mathcal{P} : \{(n_0, r_0, n_1) \rightarrow (n_1, r_1, n_2) ... \rightarrow (n_{L-1}, r_{L-1}, n_L)\} \quad (1)$$

Instance $\mathcal{P}$ is a randomized process that begins at $v_i; T_V(v_i) = n_i$, and at each iteration moves with respect to a specific sampling strategy P to the next v_j. After L iteration, the random metapath walk instance is denoted as the sequence $sg = \{< v_i, r_i, v_{i+1} > \sim P(v_i|v_{i+1}; \mathcal{P})\}_{i=0}^{L}$ following the naive sampling strategy,

$$P(v_i|v_{i-1}; \mathcal{P}) = \begin{cases} \frac{1}{|\mathcal{N}_{\mathcal{P}_{n_i}}(v_{i-1})|} & e_i \in E \text{ and } T_E(e_i) = \mathcal{P}_{r_i} \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

,where $\mathcal{N}_{\mathcal{P}_{n_i}}(v)$ denotes the neighbors of v of type n_i, and v_i denote the i^{th} node in the metapath sequence $\mathcal{P}$, where $e_i = (v_{i-1}, v_i)$. Important to mention that $\mathcal{P}_{n_i}$ and $\mathcal{P}_{r_i}$ denote the node and relation type of the i^{th} element if the metapath schema in $\mathcal{P}$, respectively.

The metapath instances are complete once $i \geq L$ and $\mathcal{P}$ are used symmetrically following Eq.2, where $T_V(n_0) = T_V(n_L)$, then the sampling strategy forces the recursiveness of the random metapath walk by $P(v_{L+1}|v_L; \mathcal{P}) = P(v_0|v_L; \mathcal{P})$, otherwise the length of the instance is insufficient to incorporate the predefined relationships in the metapath and is then omitted; this approach [5] is an extension of the random walk used in HG [9][15].

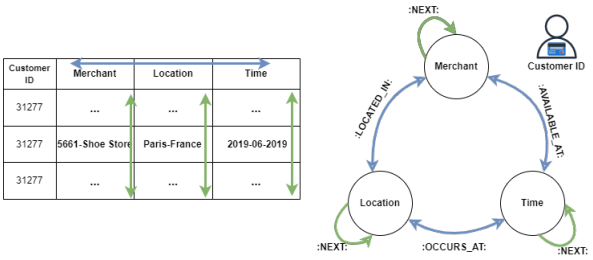

Figure 1: Credit card tabular data and the Ego-centric cardholder graph schema, green edge represent the :NEXT: transaction column-wise.

Problem Analysis and Definition for cardholder transactions. Instead of using a tripartite network to model the problem as a node-level representation [10], we propose to build an ego-centric cardholder graph dataset $\mathcal{G}$ to analyze and uncover collective hidden behavior patterns as a graph level representation problem.

Definition 3.3. **Ego-centric cardholder graph.** Given a set of graphs $\mathcal{G} = \{G_i\}_{i=1}^{n}$, G_k is the k^{th} heterogeneous ego-centric cardholder graph includes the cardholder's merchant code categories (MCC), locations and discretized time units [6] as nodes V_k,

$$V = \bigcup_{c}^{cols} V_c; \text{cols} = \{MCC, Location, Time\} \quad (3)$$

and the cardholder's transaction semantics as edges $E_k = \{(v_i, v_j) \in V_k \times V_k, w_{i,j}^k \in W_k : (v_i, v_j, w_{i,j}^k)\}$ and the cardholder's transition matrix $W_k \in [0, 1]^{V_k \times V_k}$ defined from G_k adjacency matrix denoted $A^{(k)}$, where $[A^{(k)}]_{ij}$ represent the total consecutive pairs of transactions within specific time window Δt, set 24 hours, if the node types are the same in the cardholder historical transaction records,

$$[A^{(k)}]_{ij} = \underbrace{\sum_{T_V(v_i)==T_V(v_j)} [1]_{\Delta t}}_{\text{row-wise operation}} + \underbrace{\sum_{T_V(v_i)\neq T_V(v_j)} 1}_{\text{column-wise operation}} \quad (4)$$

The transition matrix W_k is refined with a 2-hop step with a column stochastic matrix $W_k = (A_k D_k^{-1})^2$, where D_k is the diagonal matrix with $D_{ij} = \sum_{j=1}^{|V|} A_{k,(i,j)}$. We amplify the signal in historical data by refining the transition matrix as a weight matrix to enrich the topological and semantic features in G_k

To avoid any confusion, we call self-loops in the graph schema by 'next transaction' edge denoted by :NEXT:, for example the merchant-next-merchant, as shown in Fig. 1.

Problem Definition. After defining the ego-centric cardholder graph dataset $\mathcal{G}$, we want to learn the vector representations $\phi \in \mathbb{R}^{|\mathcal{G}| \times d}$ for every graph $G_i \in \mathcal{G}$, where d is the embedding vector size, $d > 0$.

Problem 1. (Learn from graph substructures) Given a graph $G_k \in \mathcal{G}$, and a set of meta-path instances sg, learn an embedding function $f_G : G_k \rightarrow h_{G_k} \in \mathbb{R}^d$ that preserves the substructure properties of G_k in the latent graph embeddings space ϕ.

Problem 2. (Avoid aggregation functions) The aggregation function operates on graph substructures to represent G_k using average, sum or max pooling layer, this technique ignores the graph topology. Given a graph $G_k \in \mathcal{G}$, learn ϕ that preserves the proximity between similar graphs in $\mathcal{G}$ and avoid explicit proxy aggregation functions that operate on latent substructure graph embedding.

4 THE PROPOSED APPROACH

Our framework is designed based on an intuitive analogy of graph-level embeddings to define a custom loss function. Thus, in this section, we first introduce our motivation and present G-HIN2VEC as an unsupervised learning framework for heterogeneous graphs.

4.1 Motivation

The intuition behind our framework is conducted by a simple analogy with graph kernels for graph level embeddings. Given two graphs G_i and G_j, for each atomic graph substructures a distribution-based measure g is quantified as the similarity between two graphs,

$$D(G_i, G_j) = \sum_{v=1}^{U} g(h_{i,v}, h_{j,v}) \quad (5)$$

where $U = \mathcal{N}(G_i) \cap \mathcal{N}(G_j)$ is the common substructures in both graphs, in this case nodes where $\mathcal{N}(G_i)$ represent nodes of G_i and $h_{i,v}$ could indicates the color of the node v in G_i for WL-OA [18], or a random graphlet starting from v for graphlet kernel [2] or a subgraph where the ego node is v for SP-kernel [19], and D is considered as a graph representation kernel. In GNNs, g is applied on atomic graph

substructures and D is considered as a graph representation,

$$D(G_i, \varnothing) = \sum_{v=1}^{U} g(h_{i,v}, \varnothing) = \sum_{v=1}^{N_{Gi}} w_v h_{i,v} \tag{6}$$

here g could be a static pooling such as average operation (where $w_v = \frac{1}{|N_{G_i}|}$) or a learned aggregation layer [31]. Regardless of the local properties encoder function D, a global properties encoder function Q could be added and applied to $\mathcal{G}$ to represent G_i by a n-dimensional vector representing the distances from all graphs,

$$Q(G_i) = \bigcup_{G_j}^{\mathcal{G}} \{D(G_i, G_j)\} \tag{7}$$

In Eq.5 and 6, g is seen as a distance function that quantifies the inter/intra graph(s) dispersion of different local D and global Q representations. Our motivation is to learn the graph representation by quantifying the graph dispersion without explicitly aggregating the graph substructures, while at the same time avoiding the use of the true distance matrix Eq.7 for global properties.

4.2 G-HIN2Vec Embedding Learning

The key novelty of G-HIN2vec is the use of intra-graph substructures to learn proximity for heterogeneous graphs, the proximity here is defined as inclusion task: do $sg_i \subseteq G_i$?, where sg_i is an element of G_i; conditioned by $\mathcal{P}$, the semantic meta-path context.

Given $\mathcal{G}$ and $\mathcal{P}^A \in \mathcal{P}$, the objective is to maximize the probability,

$$\arg\max_{\theta} \prod_{G_i \in G} \prod_{sg_i \in P^A(G_i)} Pr(sg_i^{(.)} | G_i; \theta) \tag{8}$$

where $sg_i^{(.)} = \{< v_{i,k}, r_{i,k}, v_{i,k+1} >\}_{k=0}^{L}$ is a sequence of triplets with respect to $\mathcal{P}^A$, which can be defined as a semantic substructures of G_i, and $Pr(sg_i^{(.)} | G_i; \theta)$ represents the conditional probability of having $sg_i^{(.)}$ given G_i.

To learn a graph representation $\phi \in \mathbb{R}^{|G| \times d}$, we made a simple extension to the heterogeneous skip-gram model presented in [5] to incorporate the metapath schema to learn from graph substructure by maximizing logarithmic probability,

$$\arg\max_{\theta_1, \theta_2} \sum_{G_i \in G} \sum_{sg_i \in P^A(G_i)} \sum_{t=1}^{L} log(Pr(sg_i^{(t+)} | G_i; \theta_1) Pr(sg_i^{(\neq t+)} | G_i; \theta_2)) \tag{9}$$

Here, it is assumed that the positive meta-path context instance $sg_i^{(\neq t+)} = \{< v_{i,k}, r_{i,k}, v_{i,k+1} >\}_{k=0, k\neq t}^{L}$ and the target instance $sg_i^{(t+)} = \{< v_{i,t}, r_{i,t}, v_{i,t+1} >\}$ are independent for a given G_i, where $sg_i^{(t+)}$ represents the t^{th} triplet in the meta-path sequence and $sg_i^{(\neq t+)}$ represents the entire sequence excluding the t^{th} triplet.

In general, heterogeneous graph requires more than one meta-path to capture rich semantics, for this, G-HIN2Vec expands Eq.9 to cover the entire set of predefined metapaths $\mathcal{P}$,

$$\arg\max_{\theta_1, \theta_2} \sum_{G_i \in G} \sum_{P^A \in P} \sum_{sg_i \in P^A(G_i)} \sum_{t=1}^{L} logPr(sg_i^{(t+)} | G_i; \theta_1) + logPr(sg_i^{(\neq t+)} | G_i; \theta_2) \tag{10}$$

where $Pr(sg_i^{(.)} | G_i)$ is defined as

$$\frac{exp(h^A_{sg_i^{(.)}} \cdot h_{Gi})}{\sum_{G_j \in G} \sum_{sg_j \in P^A(G_i)} exp(h^A_{sg_j^{(.)}} \cdot h_{Gj})} \tag{11}$$

To avoid considering the entire graph dataset substructures for each meta-path in $\mathcal{P}$ and all targets in each element of $sg_i^{(.)}$, we use negative sampling technique to efficiently train our model as a triplet network, where M is the negative sample instances set to 5. Inspired by [20], the sampling distribution is specified for each meta-path in $\mathcal{P}$. In the same direction, Eq.10 can be expressed as a distance-based objective function following Eq.5 and 6 to customize the loss function proposed in [7][14]. Therefore, we have the following objective:

$$\mathscr{L} = \sum_{\substack{sg_i^+ \sim P^A(G_i) \\ sg_j^- \sim P^{A-}(G_i) \\ P^A \sim P \\ sg_j^- \neq sg_i^+}} \underbrace{\left[g_{nn}(h^A_{sg_i^{t+}}, h^A_{G}i)^2 - g_{nn}(h^{A-}_{sg_j^{t-}}, h^{A-}_{G}{}_i)^2 + \alpha_1 \right]_+}_{\mathscr{L}^{Node}_{Triplet}\text{ : Triplet node loss}} + \underbrace{\left[f_{nn}(h^A_{sg_i^{\neq t+}}, h^A_{G}i)^2 - f_{nn}(h^{A-}_{sg_j^{\neq t-}}, h^{A-}_{G}{}_i)^2 + \alpha_2 \right]_+}_{\mathscr{L}^{Graph}_{Triplet}\text{ : Triplet graph loss}} \tag{12}$$

where $[\cdot]_+ = max(\cdot, 0)$, and $P^{A-} \underset{unif.}{\sim} \{P \in \mathcal{P} | P \neq \mathcal{P}^A\}$ is the negative metapath and sg_j^- is the negative metapath instance defined as a random sequence of triplets with respect to P^{A-}, sg_i^- does not necessarily exist in $\mathcal{G}$ following Algorithm 1.

Hidden substructure representations are the output of a shared weight network for each triplet element, nodes, and edge, $h_{vk} = \sigma(W_v \vec{x}_{v_k})$ and $h_{rk} = \sigma(W_r \vec{x}_{r_k})$, respectively, with $\vec{x}_{v \in \mathbb{R}^{|V|}}$ and $\vec{x}_{r \in \mathbb{R}^{|E|}}$ are the one-hot indicator vectors, and $W_v \in \mathbb{R}^{|V| \times d}$ and $W_r \in \mathbb{R}^{|E| \times d}$ are weight matrices. The same applies for the hidden representation of the graph $h_{G_i} = \sigma(W_G \vec{x}_{G_i})$, with $\vec{x}_{G_i \in \mathbb{R}^{|G|}}$ and $W_G \in \mathbb{R}^{|G| \times d}$ are the embedding and weight matrices, $\sigma(\cdot)$ is the sigmoid function.

Then a metapath-type-aware concatenation mechanism is used to stabilize the learning process with respect to the heterogeneity of metapaths as a triplet concatenation operation $h^A_{sg_i^{(.)}} = C(sg_i^{(.)})$, defined as

$$h^A_{sg_i^{(.)}} = W_{P^A} (\Big\|_{k=1}^{L} (\frac{1-\beta}{2} h_{vk}) \odot (\beta h_{rk}) \odot (\frac{1-\beta}{2} h_{vk+1})) \tag{13}$$

where $h^A_{sg_i^{(.)}} \in \mathbb{R}^c$ is the hidden representation of $sg_i^{(\cdot)}$ and $W_{P^A} \in \mathbb{R}^{d \times c}$ is a linear transformation of a nonlinear function to project

the hidden representations of the substructure to a specific output dimension, and $\beta \in [0, 1]$ is a signal amplifier for triplet relations, set to 0.4. Same for graph hidden representation where $h^A_{G_i} \in \mathbb{R}^c$,

$$h^A_{G_i} = W_{PA} h_{Gi} \tag{14}$$

This projection Eq.13 is a metapath-aware operation is used for similarity measurement, without an activation function, to map the input set of vectors to an adequate output vector space.

In summary, given the projected feature vector for G_i and $sg_i^{(\cdot)}$ positive and negative contexts, we learn two distinct functions to measure the similarity between the hidden representations of the graph and nodes, $< h^A_{sg_i^{t+}}, h^A_{G_i}, >$ and $< h^{A-}_{sg_i^{t-}}, h^{A-}_{G_i}, >$, respectively, with $g_{nn}(\cdot)$, and between the hidden representations of the graph and the metapath instance, $< h^A_{sg_i^{\neq t+}}, h^A_{G_i}, >$ and $< h^{A-}_{sg_i^{\neq t-}}, h^{A-}_{G_i}, >$, respectively, with $f_{nn}(\cdot)$. Following [23], both $g_{nn}(\cdot)$ and $f_{nn}(\cdot)$ are fully connected layers with a one-dimensional output, instead of using static functions such as Euclidean distance or cosine similarity. As $g_{nn}(\cdot)$ and $f_{nn}(\cdot)$ represents a learned metric where the largest the value more the hidden representations are dissimilar, therefore, the dissimilarity and probability defined in Eq.10 must be correlated positively with both functions. Similarly to [23], we define both functions with a softmax activation layer to normalize the output in $[0, 1]^2$ and keep only one dimension as a dissimilarity value formulated as follows:

$$\begin{cases} g_{nn}(h^A_{sg_i^{(\cdot)}}, h^A_{G}i) = softmax(W_g^T \cdot (h^A_{sg_i^{(\cdot)}}, h^A_{G}i) + B_g)_0 \\ f_{nn}(h^A_{sg_i^{(\cdot)}}, h^A_{G}i) = softmax(W_f^T \cdot (h^A_{sg_i^{(\cdot)}}, h^A_{G}i) + B_f)_0 \end{cases} \tag{15}$$

where $W_g \in \mathbb{R}^{2c\times 2}$, $W_f \in \mathbb{R}^{2c\times 2}$, $B_g \in \mathbb{R}^{2c\times 1}$, and $B_f \in \mathbb{R}^{2c\times 1}$, are the weight matrices for the learned metric function g_{nn} and f_{nn} and their bias terms, respectively.

In our loss definition Eq.12, we fixed two margins threshold terms $\alpha_1 \in [0, 1]$ and $\alpha_2 \in [0, 1]$, set to 1 and 0.6, both control the intra and inter graph variations trough the substructure encoding. we adapt the two margin terms to produce small intra and larger inter graph similarity metrics.

Algorithm 1: MetapathSeqNoising Function.

Input : $sg_i^+, \mathcal{P}^{A-}, L, \lambda$
Output : sg_i^- negative metapath instance.
$L' = \lceil \lambda L \rceil$ // # of noise edges ;
$(v_t^-, v_{t+1}^-) \leftarrow$ Shuffle($\{v \in V | T_V(v) \in P^{A-}\}, L'$);
$r_t^- \leftarrow$ Shuffle($\{r \in R | T_E(v_t^-, v_{t+1}^-) \in P^{A-}\}, L'$);
// Noise injection in sg_i^+ as concatenation process;
$L1 = \lceil \frac{L-L'}{2} \rceil$and, $L2 = \lceil \frac{L+L'}{2} \rceil$;
$sg_i^- \leftarrow \{sg^+_{i,k}\}_{k=1}^{L1} \frown \{< v_t^-, r_t^-, v_{t+1}^- >\}_{t=1}^{L'} \frown \{sg^+_{i,k}\}_{k=L2;k\leq L}^{L}$;
Return : sg_i^-

For unsupervised learning, without any graph labels, we can optimize the G-HIN2VEC weights by minimizing the double-triplet loss function through our training data preparation process to produce embedding for graph dataset, nodes, and relations denoted as follow

Algorithm 2: G-HIN2VEC training algorithm.

Input : The heterogeneous graph dataset $\mathcal{G}$,
node types $N = \{N_1, ..., N_{|N|}\}$,
edge types $R = \{R_1, ..., R_{|R|}\}$,
metapaths $P = \{P^1, ...P^{|P|}\}$,
global learnt weight W_j, batch size B
walk length L, negative samples M, noise level λ
Output : The global weight W_{i+1} matrix.
$\mathfrak{S}$ = Shuffle($\mathcal{G}, B$); // Shuffle and Random B samples.
for $i = 1, ..., B$ **do**
  $G_i = \mathfrak{S}_i$;
  $\mathcal{P}^A$ = Shuffle($\mathcal{P}, 1$);
  $\mathcal{P}^{A-}$ = Shuffle($\{P \in \mathcal{P} | P \neq \mathcal{P}^A\}, M$);
  $sg_i^+ :=$ MetapathSeq($G_i, \mathcal{P}^A, L$);
  for $m = 1, ..., M$ **do**
    // M random negative samples, set to 5.
    $sg_i^- =$ MetapathSeqNoising($sg_i^+, \mathcal{P}^{A-}, L, \lambda$);
    $h_{G_i} = \sigma(W_G \vec{x}_{G_i})$;
    for $k = 1, ..., L$ **do**
      // For each Triplet in sg_i^+ and sg_i^-;
      $< v^+_{i,k}, r^+_{i,k}, v^+_{i,k+1} > \leftarrow sg^+_{i,k}$;
      $< v^-_{i,k}, r^-_{i,k}, v^-_{i,k+1} > \leftarrow sg^-_{i,k}$;
      // Relation hidden representation transformations;
      $h_{r^+_{i,k}} = \sigma(W_r \vec{r}^+_{i,k})$;
      $h_{r^-_{i,k}} = \sigma(W_r \vec{r}^-_{i,k})$;
      // Node hidden representation transformations;
      $h_{v^+_{i,k}}, h_{v^+_{i,k+1}} = \sigma(W_v \vec{v}^+_{i,k}), \sigma(W_v \vec{v}^+_{i,k+1})$;
      $h_{v^-_{i,k}}, h_{v^-_{i,k+1}} = \sigma(W_v \vec{v}^-_{i,k}), \sigma(W_v \vec{v}^-_{i,k+1})$
    end
    t = Shuffle($[1, L], 1$)// Sample target context ;
    // Representations alignment and concatenation;
    $h^A_{G_i}, h^{A-}_{G_i} = W_{PA} h_{G_i}$, $W_{PA^-} h_{G_i}$;
    $h^A_{sg_i^{t+}}, h^{A-}_{sg_j^{t-}} = C(h_{sg_j^{t+}})$, $C(h_{sg_j^{t-}})$;
    $h^A_{sg_i^{\neq t+}}, h^{A-}_{sg_j^{\neq t-}} = C(h_{sg_j^{\neq t+}})$, $C(h_{sg_j^{\neq t-}})$;
    // Output layer as learned dissimilarity metric;
    $d^+_{node}, d^-_{node} = g_{nn}(h^A_{sg_i^{t+}}, h^A_{G_i}), g_{nn}(h^{A-}_{sg_i^{t-}}, h^{A-}_{G_i})$;
    $d^+_{mp}, d^-_{mp} = f_{nn}(h^A_{sg_i^{\neq t+}}, h^A_{G_i}), f_{nn}(h^{A-}_{sg_i^{\neq t-}}, h^{A-}_{G_i})$;
    // Cost function Eq.12;
    $Cost_i += [d^+_{node} - d^-_{node} + \alpha_1]_+ + [d^+_{mp} - d^-_{mp} + \alpha_2]_+$;
  end
end
$W_{i+1} := miniBatchSGD(W_i, Cost_i)$ // Weight updates;

$\phi \in \mathbb{R}^{n\times d}$,$X_v \in \mathbb{R}^{|V|\times d}$, and $X_r \in \mathbb{R}^{|R|\times d}$, respectively. In this work we are interested in graph level representation ϕ.

4.3 Training Data Preparation

As introduced previously, G-HIN2Vec uses the negative sampling technique to generate training data by optimizing double-triplet loss functions. Therefore, data preparation is a crucial phase in our

approach. As detailed in Algorithm 2. We train our model in mini batches, sampled from $\mathcal{G}$, then generate the positive context sg_i^+ from G_i with a predefined meta-path $\mathcal{P}^A \underset{unif.}{\sim} \mathcal{P}$ following our custom sampling strategy,

$$p(v^i|v^{i-1};\mathcal{P}) = \begin{cases} \beta \frac{1}{|\mathcal{N}_{\mathcal{P}_{n_i}}(v^{i-1})|} & e_i \in E \text{ and } T_E(e_i) = \mathcal{P}_{r_i} \\ (1-\beta)\frac{1}{|\mathcal{N}_{\neq\mathcal{P}_{n_i}}(v^{i-1})|} & e_i \in E \text{ and } T_E(e_i) \neq \mathcal{P}_{r_i} \\ 0 & e_i \notin E; \text{ where } e_i = (v^i, v^{i-1}) \end{cases} \tag{16}$$

where β is the teleportation term to escape nodes with high centrality and $\mathcal{N}_{\neq\mathcal{P}_{n_i}}(v)$ denotes the neighbor v^i of different types of nodes n_i, denoted by $MetapathSeq$ in Algorithm 1.

For better hard negative samples and instead of generating negative context sg_i^- from different graphs, we opted to noise the original positive metapath instances, sg_i^+, with respect to different metapath $P^{A-} \underset{unif.}{\sim} \{P \in \mathcal{P} | P \neq \mathcal{P}^A\}$, $\lambda \in [0,1]$ is considered as noise level and set to 0.3, This process is denoted by $MetapathSeqNoising$. After generating positive and negative metapath instances, we perform set of linear transformations to obtain the final triplet hidden representations, which can be used for dissimilarity metric estimations. Depends on the quantified metrics in our unsupervised framework, we optimize the global model weights by minimizing the double-triplet loss via back-propagation and gradient descent, to learn meaningful graph level embeddings for heterogeneous graph dataset.

5 EXPERIMENTS

In this section, we evaluate the proposed model in a real-world credit card transaction data set from an European bank. Our empirical analysis focuses on qualitative and quantitative analysis. It is important to mention that the data set used in the current research work is fully aligned with the General Data Protection Regulation in the European Union (GDPR) to protect personal data as defined in art. (4).

5.1 Datasets and Tasks

We conducted our experiments on the transaction record dataset between 2019 and 2021, the dataset has 100000 cardholder transaction data, each record has information related to Merchants (e.g. "5661-Shoe Stores"), and merchant locations (city-country, e.g. "Paris-France") and the transaction dates (e.g. "2019-06-21"). The original data set is transformed to the ego-centric cardholder graph dataset, as described in Section 3.3. On the other hand, we use cardholders socio-demographic normalized attributes for supervised machine learning experiments, where

- $Y^i_{Gender} \in \{0,1\}, \forall i \in \{1,..,n\}$ denotes i^{th} cardholder gender.
- $Y^i_{Income} \in [0,1], \forall i \in \{1,..,n\}$ denotes i^{th} cardholder salary.
- $Y^i_{Age} \in [0,1] \forall i \in \{1,..,n\}$ denotes i^{th} cardholder age.

For quantitative analysis, we performed a binary classification task on Y_{Gender} and a regression task on Y_{Age} and Y_{Income} to benchmark different hidden representation models on predictive tasks for the ego-centric graph of cardholders. On the other hand, we also performed qualitative analysis based on cardholders representations as a cluster analysis task to illustrate and interpret hidden communities.

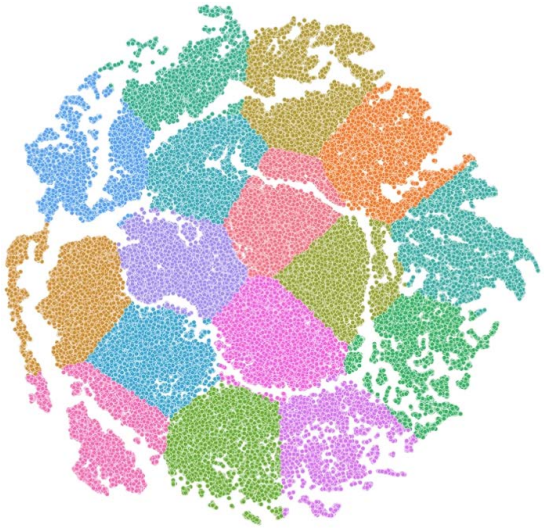

Figure 2: Cardholder embeddings t-SNE visualization of 16 identified clusters in ● Grocery Shoppers, ● Commuters ● Young Workers ● High-Tech ● Elderly ● Restaurant-goers ● Drivers ● Musicophiles ● Bookworm/Bibliophiles ● Business ● FastFood-goers ● Householders ● Cash-Only ● Gamblers ● Peripatetic/Travelers ● Epicureans

5.2 Baselines and Experimental Settings

We use the following baselines, adapted if needed, as heterogeneous graph embedding to compare G-HIN2VEC with different methods: First, **Vectorization-based methods** where the graph is represented by a unigram, bigram, and Bag-of-Features of graph substructures, and **Kernel-based methods** as a traditional graph baseline where the graph is represented as a symmetric matrix by different graph matching kernels: Graphlet kernel [2] (**GK**), Shortest path kernel[19] (**SPK**) and Weisfeiler-Lehman framework[18] (**WLG**). Finally, **GNN-based methods** where different deep learning architectures are applied on graphs: **Metapath2vec**[5], **Graph2Vec**[14], **HIN2Vec**[7], **DiffPool**[31], and **DGK**, **DSP** and **DWL** represent the "Deep" variant of kernel-based methods [29].

BQ 1 :	What spending habit does the cardholder have?
Metapath 1 :	$M \xrightarrow{:Next:} M$
BQ 2 :	Where the cardholder use the credit card ?
Metapath 2 :	$M \xrightarrow{:Located\_in:} L \xrightarrow{:Located\_in:} M$
BQ 3 :	When the cardholder use the credit card ?
Metapath 3 :	$M \xrightarrow{:Available\_at:} T \xrightarrow{:Available\_at:} M$
BQ 4 :	When and where the the credit card is used ?
Metapath 4 :	$M \xrightarrow{:Available\_at:} L \xrightarrow{:Located\_at:} T \xrightarrow{:Available\_at:} M$

Table 1: Set of business questions as predefined meta-paths by financial experts. (M: Merchant, L: Location and T: Time)

For training our model, we set the dropout rate to 0.5, and we use the same splits of training, validation, and testing sets, on the other hand, we employ the SGD optimizer with the learning rate

	Gender			Income		Age	
Method	Acc.	F1	AUC	R2	MAE	R2	MAE
LR (uni & bi-grams)	0.6133	0.6738	0.601	0.1067	0.1531	0.1045	12.7074
LR (BoF)	0.7295	0.7642	0.722	0.1751	0.1409	0.2994	10.9955
GK	0.5825	0.6636	0.5647	0.0449	0.1618	0.0846	12.5174
DGK	0.5399	0.6098	0.5185	0.0472	0.1469	0.0842	11.4422
SP	0.5477	0.6930	0.5002	0.0698	0.1597	0.0873	12.9465
DSP	0.5566	0.7133	0.5137	0.0798	0.1629	0.0922	13.1960
WL	0.7001	0.7312	0.6993	0.2030	0.141	0.4301	9.8510
DWL	0.7032	0.7538	0.7292	0.2255	0.1381	0.4717	9.5978
Metapath2vec	0.6102	0.6697	0.5586	0.1105	0.1489	0.1349	12.3272
Graph2Vec	0.7220	0.7667	0.7331	0.2394	0.1354	0.5384	9.4021
HIN2Vec	0.7809	0.8029	0.7906	0.2318	0.1399	0.6385	6.6019
DiffPool	0.8047	0.8365	0.8194	0.3088	0.1212	**0.7108**	**5.9277**
G-HIN2VEC	**0.8244**	**0.8520**	**0.8311**	**0.3310**	**0.1137**	0.6843	6.3157

Table 2: The performance of G-HIN2VEC and the baseline methods in predicting cardholder attributes.

set to 0.005 and the weight decay with L2 penalty set to 0.001 for 50 epochs. For the downstream tasks, we use linear and logistic regression on top of the graph embedding vector. On the other hand, we analyze the effectiveness of different models for graph-level representation in the classification task using averaged accuracy, AUC, and F-1 metrics. For regression tasks, the R-squared (R2) and mean absolute error (MAE) metrics are reported.

5.3 Empirical Validation

We performed empirical validation on the graph data set following the experimental settings. For non-graph level representation models, such as HIN2VEC, we added a simple averaging layer on top of each node-level representation to generate graph representations. On the other hand, in heterogeneous guided random walk-based GNNs, the definition of the metapath is required. For this purpose, experts help us to define a set of meta-paths to answer specific business questions (BQ), as shown in Table. 1. Furthermore, metapath-free baselines ignore the predefined metapaths for both node- and graph-level representation methods, and this will not be considered in both experiment analysis due to the original implementations.

5.4 Quantitative analysis

5.4.1 Regression tasks. On both income and age prediction tasks, (a) vectorization and graph kernel-based methods show poor performance in both metrics, on the other hand, (b) GNN-based baselines show slightly better performance for heterogeneous graph embedding methods, and (c) relative to DiffPool, our model is 7.18% and 6.19% more effective in R2 and MAE, respectively, for income prediction, which follows a non-Gaussian distribution, and shows a comparable performance for predicting age as Gaussian distribution. Thus, G-HIN2Vec generates desirable graph-level embeddings for regression tasks as shown for income prediction.

5.4.2 Classification task. The performance for gender classification is summarized in Table. 2. For this task, G-HIN2Vec significantly improves performance in all metrics; (a) the results highlight that vectorization methods perform better than graph kernel-based methods; on the other hand, (b) DiffPool is the best baseline model that demonstrates how graph level embedding is improved using different aggregation techniques rather than averaging at the top of the node level; (c) unlike regression tasks, G-HIN2Vec shows relative improvements compared to DiffPool in all metrics of 2.44%, 1.85% and 1.43% in Acc., F1 and AUC , respectively.

It is evident from Table. 2 that the quantitative analysis leads to a slight improvement and comparable performance. We explain this using the distributions of the graph sizes and their relative global coefficient of assortativity in $\mathcal{G}$, shown in Fig. **??**. We can clearly see that the estimated assortativity coefficient is negative, which highlights that our graphs are completely disassortative. This is considered as the main limitation of GNN models, as shown in [21], where the prediction performance of GNN models is directly affected by the assortativity in node level tasks, which can be generalized for the graph level tasks. For this reason, we added a teleportation term β in our sampling strategy Eq.16 and this finding further supports the quantitative analysis that G-HIN2VEC performs better in regression and binary classification in a disassortative graph data set.

5.5 Qualitative analysis

In this section, we perform cluster assignment task using G-HIN2VEC embedding as the HIN graph clustering task. As we show in the Figure. 2, we provide a t-SNE visualization of cardholder embeddings, our assumption is that cardholders are semantically similar in using credit cards, tend to be embedded in close proximity, as shown in [14] and [5] for graph and node level representations, respectively. Thus, two cardholders are semantically similar if their credit card usage behavior is similar with respect to a set of predefined metapaths Table.1.To better understand cardholders, we use the learned graph embedding to perform clustering, here we use k-means algorithm to cluster and find the optimal number of cluster. For this, we determined the number of clusters using the elbow method, following this approach we found 16 clusters in the graph latent space.

The 16 groups of cardholders share similarities with each other, where each has interesting behavioral preferences considering lifestyle

after extensive research on credit card analysis [4]. Therefore, the cluster identity identification is a post-grouping analysis, where we extract meaningful metapath instances randomly sampled during the training to answer "How frequently a directed sequence appear in a cluster ?". Experts are involved in this task to identify the identity of each group based on a set of sequences, where each cardholder is annotated by lifestyle category, as shown in Figure. 2.

6 CONCLUSION AND FUTURE WORK

In heterogeneous graph-level embedding, state-of-the-art methods either attempt to perform an aggregation layer on top of node-level representation or ignore the heterogeneity of graphs and apply homogeneous graph representation models. While recent methods have shown a considerable merits, to tackle this problem by using true distance matrix computationally expensive and model the problem as supervised learning, this highlight how limited to real-world applications are. Instead, G-HIN2VEC is proposed as an unsupervised framework using a double-triplet loss without ignoring the graph heterogeneity, post and during the learning phase. We adopt the perturbation negative sampling technique to avoid generating data from the entire graph dataset and remove graph-to-graph matching for a more realistic setting. We also implement a metapath-type-aware concatenation mechanism to encode metapath-type features in graph representation. All experiments are conducted on a real-world financial data set, modeled as an ego-centric graph dataset for cardholders to benchmark graph neural network models for graph-level representation, where our model performs better in different designed tasks compared to different models.

7 ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their valuable comments and feedback and the Data Analysis Lab team as well as the experts from our industry partner for their valuable knowledge sharing. This work was partly supported by the Luxembourg National Research Fund (FNR) under grant 15829274 and partly supported by ANR-20-CE39-0008.

REFERENCES

[1] Yunsheng Bai, Hao Ding, Yang Qiao, Agustin Marinovic, Ken Gu, Ting Chen, Yizhou Sun, and Wei Wang. 2019. Unsupervised inductive graph-level representation learning via graph-graph proximity. *arXiv:1904.01098* (2019).

[2] Karsten M Borgwardt and Hans-Peter Kriegel. 2005. Shortest-path kernels on graphs. In *Fifth IEEE international conference on data mining (ICDM'05)*. IEEE.

[3] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. 2018. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering* 30, 9 (2018), 1616–1637.

[4] Riccardo Di Clemente, Miguel Luengo-Oroz, Matias Travizano, Sharon Xu, Bapu Vaitla, and Marta C González. 2018. Sequences of purchases in credit card data reveal lifestyles in urban populations. *Nature communications* 9, 1 (2018), 1–8.

[5] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 135–144.

[6] Curtis E. Dyreson, William S. Evans, Hong Lin, and Richard T. Snodgrass. 2000. Efficiently supporting temporal granularities. *IEEE Transactions on Knowledge and Data Engineering* 12, 4 (2000), 568–587.

[7] Tao-yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*.

[8] Palash Goyal and Emilio Ferrara. 2018. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems* 151 (2018), 78–94.

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

[10] Anish Khazane, Jonathan Rider, Max Serpe, Antonia Gogoglou, Keegan Hines, C Bayan Bruss, and Richard Serpe. 2019. Deeptrax: Embedding graphs of financial transactions. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. IEEE, 126–133.

[11] Ziqi Liu, Chaochao Chen, Xinxing Yang, Jun Zhou, Xiaolong Li, and Le Song. 2018. Heterogeneous graph neural networks for malicious account detection. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 2077–2085.

[12] Ziqi Liu, Zhiqiang Wang, Yue Shen, Jian Ma, Wenliang Zhong, Jinjie Gu, Jun Zhou, Shuang Yang, et al. 2019. Graph representation learning for merchant incentive optimization in mobile payment marketing. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*.

[13] Guixiang Ma, Nesreen K Ahmed, Theodore L Willke, and Philip S Yu. 2021. Deep graph similarity learning: A survey. *Data Mining and Knowledge Discovery* 35, 3 (2021), 688–725.

[14] Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. 2017. graph2vec: Learning distributed representations of graphs. *arXiv preprint arXiv:1707.05005* (2017).

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

[16] Susie Xi Rao, Shuai Zhang, Zhichao Han, Zitao Zhang, Wei Min, Zhiyao Chen, Yinan Shan, Yang Zhao, and Ce Zhang. 2020. xFraud: Explainable fraud transaction detection on heterogeneous graphs. *arXiv preprint arXiv:2011.12193* (2020).

[17] Yuxiang Ren, Hao Zhu, Jiawei Zhang, Peng Dai, and Liefeng Bo. 2021. Ensemfdet: An ensemble approach to fraud detection based on bipartite graph. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 2039–2044.

[18] Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. 2011. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research* 12, 9 (2011).

[19] Nino Shervashidze, SVN Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten Borgwardt. 2009. Efficient graphlet kernels for large graph comparison. In *Artificial intelligence and statistics*. PMLR, 488–495.

[20] Lichao Sun, Lifang He, Zhipeng Huang, Bokai Cao, Congying Xia, Xiaokai Wei, and S Yu Philip. 2018. Joint embedding of meta-path and meta-graph for heterogeneous information networks. In *2018 IEEE international conference on big knowledge (ICBK)*. IEEE.

[21] Susheel Suresh, Vinith Budde, Jennifer Neville, Pan Li, and Jianzhu Ma. 2021. Breaking the limit of graph neural networks by improving the assortativity of graphs with local mixing patterns. *arXiv preprint arXiv:2106.06586* (2021).

[22] S Vichy N Vishwanathan, Nicol N Schraudolph, Risi Kondor, and Karsten M Borgwardt. 2010. Graph kernels. *Journal of Machine Learning Research* 11 (2010), 1201–1242.

[23] Faqiang Wang, Wangmeng Zuo, Liang Lin, David Zhang, and Lei Zhang. 2016. Joint learning of single-image and cross-image representations for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1288–1296.

[24] Jianian Wang, Sheng Zhang, Yanghua Xiao, and Rui Song. 2021. A review on graph neural network methods in financial applications. *arXiv preprint arXiv:2111.15367* (2021).

[25] Xiao Wang, Deyu Bo, Chuan Shi, Shaohua Fan, Yanfang Ye, and Philip S Yu. 2020. A survey on heterogeneous graph embedding: methods, techniques, applications and sources. *arXiv preprint arXiv:2011.14867* (2020).

[26] Yueyang Wang, Ziheng Duan, Binbing Liao, Fei Wu, and Yueting Zhuang. 2019. Heterogeneous attributed network embedding with graph convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33.

[27] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems* (2020).

[28] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826* (2018).

[29] Pinar Yanardag and SVN Vishwanathan. 2015. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. 1365–1374.

[30] Carl Yang, Yuxin Xiao, Yu Zhang, Yizhou Sun, and Jiawei Han. 2020. Heterogeneous network representation learning: A unified framework with survey and benchmark. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[31] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. 2018. Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems* 31 (2018).

[32] Yizhen Zheng, Vincent Lee, Zonghan Wu, and Shirui Pan. 2021. Heterogeneous Graph Attention Network for Small and Medium-Sized Enterprises Bankruptcy Prediction. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 140–151.