



HAL
open science

Graph-Level Heterogeneous Information Network Embeddings for Cardholder Transaction Analysis

Farouk Damoun, Hamida Seba, Jean Hilger, Radu State

► **To cite this version:**

Farouk Damoun, Hamida Seba, Jean Hilger, Radu State. Graph-Level Heterogeneous Information Network Embeddings for Cardholder Transaction Analysis. 2024. hal-04706259

HAL Id: hal-04706259

<https://hal.science/hal-04706259v1>

Preprint submitted on 24 Sep 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Graph-Level Heterogeneous Information Network Embeddings for Cardholder Transaction Analysis

Farouk Damoun^{1,2*}, Hamida Seba², Jean Hilger¹,
Radu State¹

¹University of Luxembourg, Luxembourg, Luxembourg.

²UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, Villeurbanne, France.

*Corresponding author(s). E-mail(s): farouk.damoun@uni.lu;
Contributing authors: hamida.seba@univ-lyon1.fr; jean.hilger@uni.lu;
radu.state@uni.lu;

Abstract

Graph-related applications, including classification, regression, and clustering, have seen significant advancements with the development of graph neural networks (GNNs). However, a gap remains in effectively using these models for heterogeneous graphs, as current methods primarily focus on homogeneous graphs, often overlooking potentially valuable semantic information. To address this issue, our work introduces a novel approach, G-HIN2VEC (Graph-Level Heterogeneous Information Network to Vector), specifically designed to generate heterogeneous graph representations. This method uniquely leverages a single graph to learn its own embeddings without relying on a graph dataset, by sharing model parameters across the dataset. Inspired by recent developments in unsupervised learning in natural language processing, G-HIN2VEC employs a negative sampling technique to learn graph-level embedding matrices from a variety of metapaths. This approach has been applied to real-world credit card data, facilitating the analysis of cardholder transactions through three downstream applications—graph-level regression and classification tasks, including age and income prediction, and gender classification. G-HIN2VEC outperforms traditional methods, demonstrating improvements in gender classification accuracy by 2.45% and income prediction R-squared (R²) by 7.19%. Furthermore, for age prediction, we achieved an increase of 6.55% in the mean absolute error (MAE) compared to DiffPool, a strong baseline.

Keywords: Unsupervised learning, Learning latent representations, Graph Neural Networks, Heterogeneous Information Network, Financial Transaction Analysis

33 **1 Introduction**

34 Financial institutions manage a variety of transaction types, encompassing a spectrum
35 of financial assets, such as loans, insurance, and investments. In today’s increasingly
36 digital era, banks, as ubiquitous entities, offer several financial card products that
37 dominate the retail banking sector. The past decade has seen an impressive increase
38 in cashless payments through plastic and virtual cards, leading to exponential growth
39 in the digital footprint of card transactions ¹.

40
41 Analyzing customer transactions is pivotal for various banking applications, includ-
42 ing but not limited to behavior modeling, product recommendation, and advertising
43 strategies [1]. Recent research in financial data analysis [2, 3] suggests the utility of
44 graph data structures to encapsulate the topological information inherent in tabu-
45 lar data, thereby extending and enriching available customer data. Currently, deep
46 learning algorithms have been refined and enhanced, expanding their applicability to
47 a wider range of problems [4–6].

48
49 Despite the advantages, graph data structures introduce additional complexity, with
50 each graph component characterized by unique attributes and local topology. Tasks
51 such as graph classification, clustering, and regression, which require vectorial repre-
52 sentations of graphs for the application of machine learning algorithms, become more
53 challenging. Techniques to overcome this challenge include Graph Kernel algorithms
54 [7], acting as hand-crafted feature extractors, and embedding algorithms that provide
55 dense latent graph encoding [8].

56
57 However, most of these embedding techniques have been designed for homogeneous
58 networks [8], and therefore their application is limited to graphs with a single type
59 of nodes and edges. However, real-world networks often comprise heterogeneous
60 information networks (HINs), where there are different types of interaction between
61 graph components. Graph Neural Networks (GNNs) offer potential solutions for such
62 graphs. GNNs have proven their effectiveness in various graph-related tasks, but they
63 often fail to preserve the network structure fully, especially for HINs [9, 10]. Graph
64 analysis tasks are often confined by the characteristics of the problem at hand and
65 the applied embedding techniques [11]. In scenarios involving node-level tasks such as
66 node regression and classification, the embeddings are typically designed to portray
67 low-dimensional representations of nodes. For example, the classification of scientific
68 publications using citation networks exemplifies a node-level task. Conversely, for
69 graph-level tasks, such as graph classification and regression, the embeddings provide
70 a low-dimensional representation of the entire graph. Examples of these tasks include
71 prediction of protein functions using chemical compound graphs and detection of
72 malware based on call graphs [12, 13].

73
74 Recently, random walk-based embedding methods have displayed significant per-
75 formance improvements in various homogeneous graph (HG) downstream tasks by

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

76 preserving the network structure [8]. However, their utility is considerably limited
77 when applied to a heterogeneous schema. Concurrently, much of the existing research
78 on graph embedding remains heavily focused on learning node representations, leav-
79 ing a gap in the comprehensive understanding and handling of complex heterogeneous
80 networks.

81

82 In this work, our neural network architecture, called G-HIN2Vec, is specifically
83 designed to embed entire heterogeneous graphs. It relies on joint learning architec-
84 tures [14] and random walk-based methods for node embedding. We evaluate the
85 quality of the generated heterogeneous graph embeddings across various downstream
86 tasks using different metrics for credit card transactions. Based on a preliminary lit-
87 erature review, we carefully selected our baseline methods. The experimental results
88 show that G-HIN2VEC embeddings outperform the baseline methods HIN2VEC [15]
89 and Metapath2Vec [16]. In particular, G-HIN2VEC exhibits a 2.45% improvement
90 in the precision of the gender classification and a 7.19% increase in the R-squared
91 income prediction (R2) over the strongest baselines, with a loss in the Mean Absolute
92 Error (MAE) in age prediction by 6.55% compared to DiffPool [17]. This shows the
93 effectiveness of our approach in enhancing customer understanding in retail banking.
94 Our contributions are as follows.

- 95 • We propose an ego-centric graph model for cardholder transactions to build an
96 unlabeled heterogeneous graph dataset inspired by egocentric thinking.
- 97 • We introduce G-HIN2Vec, a new unsupervised representation learning method
98 that employs a double-triplet loss function as task-agnostic approach for entire
99 heterogeneous graph representations.
- 100 • We experimentally benchmark various GNN models against G-HIN2VEC on
101 real-world credit card datasets for multiple downstream tasks, including graph
102 classification, regression, and clustering tasks.

103 2 Related Work

104 This section reviews three primary research lines for our work, focusing on hetero-
105 geneous information networks, graph-level embedding, and financial card transaction
106 data analysis.

- 107 • **Heterogeneous Information Network (HIN) Embedding:** Heterogeneous
108 Information Network Embedding has emerged as a significant area of research, pri-
109 marily due to its ability to effectively represent complex real-world data. These
110 networks encompass interactions among various types of objects, offering a more
111 detailed representation of real-world scenarios.

112 Recent algorithms in HIN embedding primarily focus on node or edge-level embed-
113 ding, as evidenced in studies by [15, 16, 18]. This focus, however, introduces
114 constraints when addressing graph-level downstream tasks. In a recent compre-
115 hensive review [19], 13 HIN embedding methods were benchmarked on four distinct
116 datasets designed for node- and edge-level tasks. These studies have consistently
117 demonstrated that the effectiveness of an HIN embedding algorithm depends on

118 the specific requirements of the downstream task, underlying the graph schema,
119 and other experimental variables, as highlighted in [9, 10]. A critical limitation in
120 the field is the lack of a unified framework for HIN embeddings, as pointed out
121 by [8]. This absence has been identified as a significant barrier, particularly when
122 addressing graph-level downstream tasks.

123 • **Graph-level Embedding:** Despite the abundance of research on node, edge, or
124 subgraph embedding algorithms, studies on entire graph embedding algorithms
125 are relatively sparse [12]. This impacts the downstream graph-related tasks for
126 HIN networks such as graph classification, regression, and clustering. Conversely,
127 HG embeddings have been extensively studied and are well-established in various
128 downstream tasks [20]. Both graph types embedding algorithms seek to preserve
129 topological and semantic similarity across a graph dataset. As discussed in [12] and
130 due to lack of HIN graph-level embedding algorithms, researchers often resort to
131 transforming HINs into HG datasets. This transformation is a proxy approach to
132 leverage existing homogeneous graph embedding algorithms, such as Graph2Vec
133 [20] and Deep Graph Kernels (DGK) [21], for generating graph-level embeddings.
134 While this method has led to improvements in performance, it unfortunately dis-
135 cards potentially valuable semantic information. This limitation has pushed research
136 towards developing embedding models tailored for HINs. UGraphEmb [22] addresses
137 the issue as a supervised training model that relies on existing heuristics for graph
138 similarity [23, 24], which is computationally intensive and not scalable. Instead of
139 proxy or supervised approaches, HIN representation can be generated via aggregate
140 functions, such as the average layer, to transform node/edge embeddings into graph
141 embeddings, which often leads to suboptimal results [25]. This is due to the absence
142 of unsupervised HIN graph-level embedding algorithms.

143 • **Financial Card Transactions:** While the field of card transaction research focuses
144 primarily on fraud detection, current graph-based methodologies predominantly uti-
145 lize simple graphs [2]. Due to the heterogeneity of the data, this approach does not
146 fully capture the intrinsic complexity of financial networks. This limitation high-
147 lights the growing need for HIN-based GNNs for financial transactions. For instance,
148 HINs have been used for the detection of malicious Alipay accounts as a node-level
149 task [13], risky transactions in a B2B network [26], and for bankruptcy prediction in
150 credit risk assessment [27]. However, beyond fraud detection, card transaction data
151 is being increasingly leveraged to gain insights into transaction entities. For exam-
152 ple, techniques such as text compression have been used to understand the lifestyles
153 of cardholders [1]. In addition, graph-based approaches have facilitated the analysis
154 of complex relationships between cardholders and merchants as a node-level task [3],
155 and the refinement of marketing strategies as a node-level task [28]. These diverse
156 applications underscore the versatility of HINs in unraveling complex interactions
157 within heterogeneous financial networks, as comprehensively reviewed in [29].

3 Problem Definition

Throughout this section, we introduce the notions of HIN and random metapath walks and discuss the problem addressed in this work. Table 1 shows the mathematical symbols and their definitions used in this article.

Symbol	Definition
\mathbb{G}	Dataset of all ego-centric cardholder graphs.
G_k	The k^{th} ego-centric cardholder graph in \mathbb{G} .
$\mathcal{N}_G, \mathcal{R}_G$	Set of nodes and edges in G .
N_G, E_G	Set of node and edge types in G .
τ_N, τ_E	Type mapping functions for nodes and edges.
A_k, D_k, W_k	Adjacency, diagonal, and transition matrices of G_k .
\mathcal{P}, p, π_{mp}	Predefined set of metapaths, a metapath, and its instance.
γ	Teleportation parameter used during random walks.
M	Quantity of negative samples generated during training.
ϕ	Learned graph-level embedding vectors for \mathbb{G} .
\mathbf{h}	Hidden state representation derived from the model.
α_1, α_2	Margins used in the double-triplet loss function.
θ	Set of all parameters in the G-HIN2VEC model.
W_v, W_e, W_G	weight matrices learned for nodes, edges, and graphs.

Table 1: Mathematical Symbols and Definitions

Definition 1. Heterogeneous Information Network. *Heterogeneous Information Network, denoted HIN, is defined as a network $G = (\mathcal{N}_G, \mathcal{R}_G, \tau_N, \tau_E, W)$ where G is a weighted directed graph, where \mathcal{N}_G and \mathcal{R}_G denote the set of nodes and edges, $W \in \mathbb{R}^{|\mathcal{N}_G| \times |\mathcal{N}_G|}$ is the weight matrix, and τ_N and τ_E are type mapping functions for nodes and edges, respectively. In HINs, node types are represented by $\tau_N : \mathcal{N}_G \rightarrow N_G$, where N_G is a set of node types, and the relations between nodes are indicated by $\tau_E : \mathcal{R}_G \rightarrow E_G$, where E_G is a set of edge types.*

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

Definition 2. Random Metapath Walk. *Given the schema of G as defined in Def. 1, a metapath $p \in \mathcal{P}$ is defined as the sequence of triplets (n_i, e_i, n_{i+1}) where $n_i \in N_G$ are the source and target node types, and $e \in E_G$ is the edge type, p is denoted as follows:*

$$p : \{(n_0, e_0, n_1) \rightarrow (n_1, e_1, n_2) \dots \rightarrow (n_{l-1}, e_{l-1}, n_l)\} \quad (1)$$

An instance $\pi_{mp} \sim p(G)$ is a randomized process that begins at v_i ; $\tau_N(v_i) = n_i$, and at each iteration moves with respect to a specific sampling strategy $Pr(\cdot)$, to the next v_j . After l iterations, the random metapath walk instance is denoted as the sequence $\pi_{mp} = \{ \langle v_i, e_i, v_{i+1} \rangle \sim Pr(v_i | v_{i-1}; p \in \mathcal{P}) \}_{i=0}^l$ following the naive sampling strategy, defined as a uniform random walk constrained by neighboring node types according to the metapath schema, similar to the approaches in [30, 31]. This ensures that each step of the walk follows the heterogeneous relationships defined by the network schema.

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

171 where $\mathcal{N}_{n_i}(v)$ denotes the neighbors of v of type n_i , and v_i denotes the node i^{th} in
172 the metapath sequence and $e_i = (v_{i-1}, v_i)$. It is important to mention that n_i and r_i
173 denote the node and relation type of the i^{th} element in the metapath schema in p ,
174 respectively. A metapath instance π_{mp} is completed after l iterations, ensuring that
175 the start and end node types match ($\tau_N(n_0) = \tau_N(n_l)$), as required by the schema.
176 This symmetry signifies a structured path through the network, following the sampling
177 strategy of Equation 2. If the path length l allows, the sequence is considered valid
178 and reflects the relationships defined by the metapath p ; otherwise, it's discarded as an
179 incomplete. The recursive aspect of the walk, $Pr(v_{l+1}|v_l; p \in \mathcal{P}) = Pr(v_0|v_l; p \in \mathcal{P})$,
180 allows the process to be continuous, maintaining the integrity of the network structure.
181 This methodology is common in HG [30, 31] and adapted for heterogeneous graphs in
182 [16].

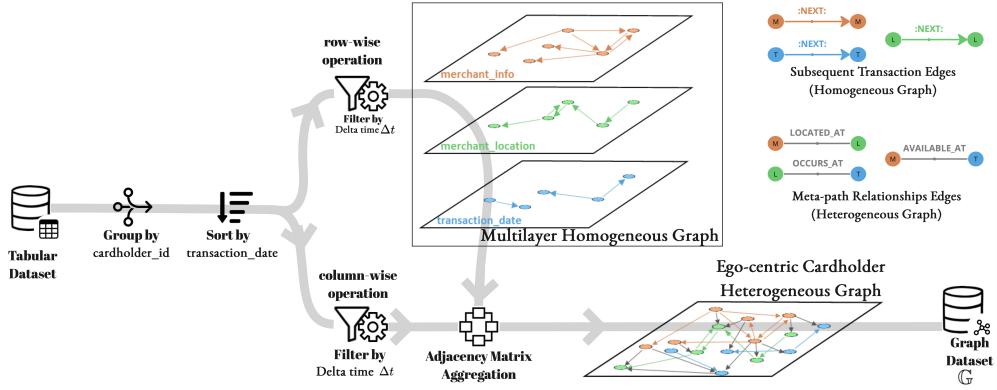


Fig. 1: Illustration of transforming credit card transaction data into an ego-centric cardholder heterogeneous graph dataset, illustrating the process of layering multiple graphs and metapaths to encapsulate the diverse financial interaction patterns within the network.

183 **Problem Analysis and Definition for Cardholder Transactions.** Instead
184 of using a tripartite large transaction network to model the problem as a node-level
185 representation [3], we propose to build an ego-centric cardholder graph dataset \mathbb{G}
186 to analyze and uncover collective hidden behavior patterns as a graph-level representation
187 problem.

Definition 3. Ego-centric cardholder graph. Given a set of graphs $\mathbb{G} = \{G_i\}_{i=1}^n$, G_k is the k^{th} heterogeneous ego-centric cardholder graph, including merchant names and code categories (MCC), locations, and discretized time units [32], as nodes as nodes $\mathcal{N}_{G_k} \subset \mathcal{N}_G$,

$$\mathcal{N}_G = \bigcup_c^{\text{cols}} \mathcal{N}_{G_c} \quad (3)$$

The term 'cols' refers specifically to the following attributes within the transaction data: `merchant_info`, `merchant_location`, and `transaction_date`. A detailed description of these fields is provided in Section 5.1. The cardholder's transaction is the edge $\mathcal{R}_{G_k} = \{(v_i, v_j) \in \mathcal{N}_{G_k} \times \mathcal{N}_{G_k}, w_{i,j}^k \in W_k : (v_i, v_j, w_{i,j}^k)\}$ and the cardholder's transition matrix $W_k \in [0, 1]^{V_k \times V_k}$ is defined from the adjacency matrix G_k denoted A_k , where $[A_k]_{ij}$ represents the total consecutive pairs of transactions within a specific time window Δt , set to 24 hours, if the node types are the same in the cardholder's historical transaction records,,

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

188 The transition matrix W_k is refined with a 2-hop step with a column stochastic
 189 matrix $W_k = (A_k D_k^{-1})^2$, where D_k is the diagonal matrix with $D_{ij} = \sum_{j=1}^{|\mathcal{N}_{G_k}|} A_{k,(i,j)}$.
 190 We amplify the signal in historical data by refining the transition matrix as a weight
 191 matrix to enrich the topological and semantic features in G_k . To avoid any confusion,
 192 we refer to self-loops in the graph schema as 'next transaction' edges, denoted by
 193 `:NEXT:`, as shown in Figure 2.

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

197
 198 **Problem 1. (Learn from graph substructures)** Given a graph $G_k \in \mathbb{G}$, and a
 199 set of metapath instances $\pi_{mp} \sim p$, learn an embedding function $f_G : G_k \rightarrow \mathbf{h}_{G_k} \in \mathbb{R}^d$
 200 that preserves the substructure properties of G_k in the latent graph embeddings
 201 space ϕ .

202
 203 **Problem 2. (Avoid aggregation functions)** The aggregation function operates on
 204 graph substructures to represent G_k using average, sum, or max pooling layers; this
 205 technique ignores the graph topology. Given a graph $G_k \in \mathbb{G}$, learn ϕ that preserves
 206 the proximity between similar graphs in \mathbb{G} and avoid explicit proxy aggregation
 207 functions that operate on latent substructure graph embedding.

208 4 The proposed approach

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

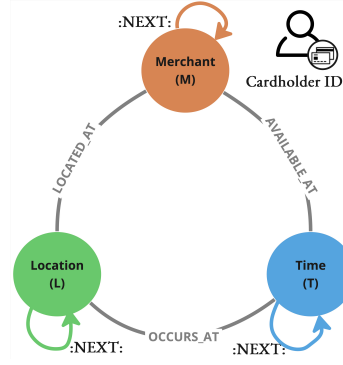


Fig. 2: Ego-centric cardholder heterogeneous graph schema

212 **4.1 Motivation**

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

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

where $U = \mathcal{N}(G_i) \cap \mathcal{N}(G_j)$ is the set of common substructures in both graphs. In this case nodes where $\mathcal{N}(G_i)$ represent nodes of G_i and $\mathbf{h}_{i,v}$ could indicates the color of the node v in G_i for WL-OA [33], or a random graphlet starting from v for graphlet kernel [23] or a subgraph where the ego node is v for SP-kernel [24], and D is considered as a graph representation kernel. In GNNs, g is applied to atomic graph substructures and D is considered as a graph representation,

$$D(G_i, \emptyset) = \sum_{v=1}^U g(\mathbf{h}(i, v), \emptyset) = \sum_{v=1}^{N_{G_i}} w_v \mathbf{h}(i, v) \quad (6)$$

Here, g could be a static pooling operation such as average operation (where $w_v = \frac{1}{|N_{G_i}|}$) or a learned aggregation layer [17]. Regardless of the local properties encoder function Q could be added and applied to \mathbb{G} to represent G_i by an n-dimensional vector representing the distances from all graphs.

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

213 In Equation 5 and 6, g is seen as a distance function that quantifies the dispersion
 214 within and between graphs for different local D and global Q representations. Our
 215 goal is to learn the graph representation by quantifying this dispersion without the
 216 need for explicit aggregation of the graph substructures. Simultaneously, our goal is
 217 to avoid using the true distance matrix (Equation 7) for global properties.

218 **4.2 G-HIN2VEC Embedding Learning**

The key novelty of G-HIN2VEC is to determine if a metapath instance π_{mp} from a given semantic context $p \in \mathcal{P}$ is a subset of the heterogeneous graph G_i , “Does $\pi_{\text{mp}} \subseteq G_i$?”, thereby learning graph proximity through intra-graph substructures.

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

$$\arg \max_{\theta} \prod_{G_i \in \mathbb{G}} \prod_{\pi_{\text{mp}} \in p^+(G_i)} Pr(\pi_{\text{mp}_i}^{(\cdot)} | G_i; \theta) \quad (8)$$

where $\pi_{\text{mp}}^{(\cdot)} = \{ \langle v_{i,k}, e_{i,k}, v_{i,k+1} \rangle \}_{k=0}^l$ is a sequence of triplets with respect to p^+ , which can be defined as a semantic substructures of G_i , and $Pr(\pi_{\text{mp}}^{(\cdot)}|G_i; \theta)$ represents the conditional probability of having $\pi_{\text{mp}}^{(\cdot)}$ given G_i .

To learn a graph representation $\phi \in \mathbb{R}^{|\mathbb{G}| \times d}$, we made a simple extension to the heterogeneous skip-gram model presented in [16], to incorporate the metapath schema to learn from graph substructure by maximizing logarithmic probability,

$$\arg \max_{\theta_1, \theta_2} \sum_{G_i \in \mathbb{G}} \sum_{\pi_{\text{mp}} \in p^+(G_i)} \sum_{t=1}^l \log(Pr(\pi_{\text{mp}}^{(t+)}|G_i; \theta_1) Pr(\pi_{\text{mp}}^{(\neq t+)}|G_i; \theta_2)) \quad (9)$$

Here, it is assumed that the positive metapath context instance $\pi_{\text{mp}}^{(\neq t+)} = \{ \langle v_{i,k}, e_{i,k}, v_{i,k+1} \rangle \}_{k=0, k \neq t}^l$ and the target instance $\pi_{\text{mp}}^{(t+)} = \{ \langle v_{i,t}, e_{i,t}, v_{i,t+1} \rangle \}$ are independent for a given G_i , where $\pi_{\text{mp}}^{(t+)}$ represents the t^{th} triplet in the metapath sequence and $\pi_{\text{mp}}^{(\neq t+)}$ represents the entire sequence excluding the t^{th} triplet.

In general, a heterogeneous graph requires more than one metapath to capture rich semantics; for this, G-HIN2VEC expands Equation 9 to cover the entire set of predefined metapaths \mathcal{P} ,

$$\arg \max_{\theta_1, \theta_2} \sum_{G_i \in \mathbb{G}} \sum_{p^+ \in \mathcal{P}} \sum_{\pi_{\text{mp}} \in p^+(G_i)} \sum_{t=1}^l \log Pr(\pi_{\text{mp}}^{(t+)}|G_i; \theta_1) + \log Pr(\pi_{\text{mp}}^{(\neq t+)}|G_i; \theta_2) \quad (10)$$

where $Pr(\pi_{\text{mp}}^{(\cdot)}|G_i)$ is defined as

$$\frac{\exp(\mathbf{h}_{\pi_{\text{mp}}^{(\cdot)}}^+ \cdot \mathbf{h}_{G_i})}{\sum_{G_j \in \mathbb{G}} \sum_{\pi_{\text{mp}_j} \in p^+(G_j)} \exp(\mathbf{h}_{\pi_{\text{mp}_j}^{(\cdot)}}^+ \cdot \mathbf{h}_{G_j})} \quad (11)$$

In order to train our G-HIN2VEC model more efficiently and avoid the exhaustive computation of considering every substructure within the entire graph dataset for each metapath in \mathcal{P} and target element in $\pi_{\text{mp}}^{(\cdot)}$, we adopt a negative sampling technique. Specifically, instead of using negative samples from different graphs in the dataset, we generate negative samples directly within the graph of interest using Algorithm 1. This allows us to create more contextually relevant and computationally efficient negative instances. For every positive instance, we generate 5 negative instances, setting the M negative sample instances to 5 for each positive instance, thus maintaining a 1:5 ratio. This ratio is a tunable hyperparameter, and while our current infrastructure supports a 1:5 ratio, this can be adjusted according to the scale and capacity of the training setup. The sampling distribution for each metapath in \mathcal{P} is carefully specified, drawing inspiration from [14]. In the same direction, Equation 10 can be expressed as

a distance-based objective function following Equation 5 and 6 to customize the loss function introduced in [15][20]. Therefore, we have the following objective:

$$\mathcal{L} = \sum_{\substack{\pi_{\text{mp}}^+ \sim p^+(G_i) \\ \pi_{\text{mp}}^- \sim p^-(G_i) \\ (p^+, p^-) \sim \mathcal{P} \\ p^+ \neq p^-}} \underbrace{\left[g_{nn}(\mathbf{h}_{\pi_{\text{mp}}^+}^+, \mathbf{h}_{G_i}^+) - g_{nn}(\mathbf{h}_{\pi_{\text{mp}}^-}^-, \mathbf{h}_{G_i}^-) + \alpha_1 \right]}_{\substack{\text{Node} \\ \text{Triplet} : \text{Triplet node loss}}}_+ + \underbrace{\left[f_{nn}(\mathbf{h}_{\pi_{\text{mp}}^+}^+, \mathbf{h}_{G_i}^+) - f_{nn}(\mathbf{h}_{\pi_{\text{mp}}^-}^-, \mathbf{h}_{G_i}^-) + \alpha_2 \right]}_{\substack{\text{Graph} \\ \text{Triplet} : \text{Triplet graph loss}}}_+ \quad (12)$$

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

Hidden substructure representations are the output of a shared weight network for each triplet element, nodes and edge, $\mathbf{h}_{v_k} = \sigma(W_v \vec{x}_{v_k})$ and $\mathbf{h}_{e_k} = \sigma(W_e \vec{x}_{e_k})$, respectively, with $\vec{x}_v \in \mathbb{R}^{|\mathcal{N}_G|}$ and $\vec{x}_e \in \mathbb{R}^{|\mathcal{E}|}$ are the one-hot indicator vectors, and $W_v \in \mathbb{R}^{|\mathcal{N}_G| \times d}$ and $W_e \in \mathbb{R}^{|\mathcal{E}| \times d}$ are weight matrices. The same applies for the hidden representation of the graph $\mathbf{h}_{G_i} = \sigma(W_G \vec{x}_{G_i})$, with $\vec{x}_{G_i} \in \mathbb{R}^{\mathbb{G}}$ and $W_G \in \mathbb{R}^{\mathbb{G} \times d}$ 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 $\mathbf{h}_{\pi_{\text{mp}}^{(\cdot)}} = \text{Concat}(\pi_{\text{mp}}^{(\cdot)})$, defined as

$$\mathbf{h}_{\pi_{\text{mp}}^{(\cdot)}} = W_p \left(\prod_{k=1}^l \left(\frac{1-\beta}{2} \mathbf{h}_{v_k} \odot (\beta \mathbf{h}_{e_k}) \odot \left(\frac{1-\beta}{2} \mathbf{h}_{v_{k+1}} \right) \right) \right) \quad (13)$$

219 where $\mathbf{h}_{\pi_{\text{mp}}^{(\cdot)}} \in \mathbb{R}^c$ is the hidden representation of $\pi_{\text{mp}}^{(\cdot)}$ and $W_p \in \mathbb{R}^{d \times c}$ is a linear
 220 transformation of a nonlinear function to project the hidden representations of the
 221 substructure to a specific output dimension, and $\beta \in [0, 1]$ is a signal amplifier for
 222 triplet relations, set to 0.4. Same for graph hidden representation where $W_p \mathbf{h}_{G_i} \in \mathbb{R}^c$.
 223 Equation 13 is a metapath-aware operation used for similarity measurement, without
 224 an activation function.

225 In summary, given the projected feature vector for G_i and $\pi_{\text{mp}}^{(\cdot)}$ positive and nega-
 226 tive contexts, we learn two distinct functions to measure the similarity between the
 227 hidden representations of the graph and nodes, $\langle \mathbf{h}_{\pi_{\text{mp}}^+}^+, \mathbf{h}_{G_i}^+ \rangle$ and $\langle \mathbf{h}_{\pi_{\text{mp}}^-}^-, \mathbf{h}_{G_i}^- \rangle$,
 228 respectively, with $g_{nn}(\cdot)$, and between the hidden representations of the graph and the
 229 metapath instance, $\langle \mathbf{h}_{\pi_{\text{mp}}^+}^+, \mathbf{h}_{G_i}^+ \rangle$ and $\langle \mathbf{h}_{\pi_{\text{mp}}^-}^-, \mathbf{h}_{G_i}^- \rangle$, respectively, with $f_{nn}(\cdot)$.
 230 Following [34], both $g_{nn}(\cdot)$ and $f_{nn}(\cdot)$ are fully connected layers with a one-dimensional
 231 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 Equation 10 must be correlated positively with both functions. Similarly to [34], 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:

$$\begin{cases} g_{nn}(\mathbf{h}_{\pi_{mp}^+}^+, \mathbf{h}_{G_i}^+) = \text{softmax}(W_g^T \cdot (\mathbf{h}_{\pi_{mp}^+}^+, \mathbf{h}_{G_i}^+) + B_g)_0 \\ f_{nn}(\mathbf{h}_{\pi_{mp}^+}^+, \mathbf{h}_{G_i}^+) = \text{softmax}(W_f^T \cdot (\mathbf{h}_{\pi_{mp}^+}^+, \mathbf{h}_{G_i}^+) + B_f)_0 \end{cases} \quad (14)$$

232 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.

234

Algorithm 1: MetapathSeqNoising Function.

Input : π_{mp}^+ , p^- , l , λ
Output : π_{mp}^- negative metapath instance.
 $l' = \lceil \lambda l \rceil$; // Number of noise edges ;
 $(v_t^-, v_{t+1}^-) \leftarrow \text{Shuffle}(\{v \in V | \tau_N(v) \in p^-\}, l')$;
235 $e_t^- \leftarrow \text{Shuffle}(\{e \in E | \tau_E(v_t^-, v_{t+1}^-) \in p^-\}, l')$;
// Noise injection in π_{mp}^+ as a concatenation process;
 $l_1 = \lceil \frac{l-l'}{2} \rceil$ and $l_2 = l - l_1$;;
 $\pi_{mp}^- \leftarrow \left\{ \pi_{i,k}^+ \right\}_{k=1}^{l_1} \oplus \{ \langle v_t^-, e_t^-, v_{t+1}^- \rangle \}_{t=1}^{l'} \oplus \left\{ \pi_{i,k}^+ \right\}_{k=l_2+1}^l$;
Return : π_{mp}^-

236

237 In our loss definition in Equation 12, we employ two margin threshold terms, α_1
238 and $\alpha_2 \in [0, 1]$, to modulate the similarity and dissimilarity within and between
239 graph representations. Specifically, α_1 , set to 1, regulates intra-graph similarity,
240 ensuring that positive samples of nodes and substructures within the same graph are
241 embedded closely together in the embedding space. On the other hand, α_2 , set to
242 0.6, governs inter-graph dissimilarity, ensuring that negative samples of nodes and
243 substructures are sufficiently separated in the embedding space. In the unsupervised
244 learning context, where graph labels are unavailable, we refine the weights of G-
245 HIN2VEC by minimizing a double-triplet loss function. This process is part of our
246 training data preparation, which aims to generate embeddings for the graph dataset,
247 nodes, and relations. These embeddings are denoted as $\phi \in \mathbb{R}^{|\mathcal{G}| \times d}$ for the graph-level,
248 $X_v \in \mathbb{R}^{|\mathcal{N}_{\mathcal{G}}| \times d}$ for nodes, and $X_r \in \mathbb{R}^{|\mathcal{R}_{\mathcal{G}}| \times d}$ for relations. Our focus in this work is on
249 the representation at the graph-level ϕ .

Algorithm 2: G-HIN2VEC training algorithm.

Input : The heterogeneous graph dataset \mathbb{G} ,
metapaths $\mathcal{P} = \{p_1, \dots, p_k\}; k > 0$,
global learned weight W_j , batch size B
walk length l , negative samples M , noise level λ

Output : The global weight W_{i+1} matrix.

$G = \text{Shuffle}(\mathbb{G}, B)$; // Shuffle and Random B samples.

for $i = 1, \dots, B$ **do**

$p^+ = \text{Shuffle}(p, 1)$;
 $p^- = \text{Shuffle}(\{p \in \mathcal{P} | p \neq p^+\}, M)$;
 $\pi_{\text{mp}}^+ := \text{MetapathSeq}(G_i, p^+, l)$;

for $m = 1, \dots, M$ **do**

// M random negative samples, set to 5.
 $\pi_{\text{mp}}^- = \text{MetapathSeqNoising}(\pi_{\text{mp}}^+, p^-, l, \lambda)$;

$\mathbf{h}_{G_i} = \sigma(W_G \vec{x}_{G_i})$;

for $k = 1, \dots, l$ **do**

// For each Triplet in π_{mp}^+ and π_{mp}^- ;
 $\langle v_{i,k}^+, e_{i,k}^+, v_{i,k+1}^+ \rangle \leftarrow \pi_{i,k}^+$;
 $\langle v_{i,k}^-, e_{i,k}^-, v_{i,k+1}^- \rangle \leftarrow \pi_{i,k}^-$;

// Relation hidden representation transformations;
 $\mathbf{h}_{e_{i,k}^+} = \sigma(W_r \vec{e}_{i,k}^+)$;
 $\mathbf{h}_{e_{i,k}^-} = \sigma(W_r \vec{e}_{i,k}^-)$;

// Node hidden representation transformations;
 $\mathbf{h}_{v_{i,k}^+}, \mathbf{h}_{v_{i,k+1}^+} = \sigma(W_v \vec{v}_{i,k}^+), \sigma(W_v \vec{v}_{i,k+1}^+)$;
 $\mathbf{h}_{v_{i,k}^-}, \mathbf{h}_{v_{i,k+1}^-} = \sigma(W_v \vec{v}_{i,k}^-), \sigma(W_v \vec{v}_{i,k+1}^-)$;

end

$t = \text{Shuffle}(\text{interval}[1, l], 1)$ // Sample target context ;

// Representations alignment and concatenation;
 $\mathbf{h}_{G_i}^+ = W_{p^+} \mathbf{h}_{G_i}$; $\mathbf{h}_{G_i}^- = W_{p^-} \mathbf{h}_{G_i}$;

$\mathbf{h}_{\pi_{\text{mp}}^+}^{t+} = \text{Concat}(\mathbf{h}_{\pi_{\text{mp}}^+}^{t+})$; $\mathbf{h}_{\pi_{\text{mp}}^-}^{t-} = \text{Concat}(\mathbf{h}_{\pi_{\text{mp}}^-}^{t-})$;

$\mathbf{h}_{\pi_{\text{mp}}^+}^{\neq t+} = \text{Concat}(\mathbf{h}_{\pi_{\text{mp}}^+}^{\neq t+})$; $\mathbf{h}_{\pi_{\text{mp}}^-}^{\neq t-} = \text{Concat}(\mathbf{h}_{\pi_{\text{mp}}^-}^{\neq t-})$;

// Output layer as learned dissimilarity metric;
 $d_{\text{node}}^+, d_{\text{node}}^- = g_{nn}(\mathbf{h}_{\pi_{\text{mp}}^+}^{t+}, \mathbf{h}_{G_i}^+)^2, g_{nn}(\mathbf{h}_{\pi_{\text{mp}}^-}^{t-}, \mathbf{h}_{G_i}^-)^2$;

$d_{\text{mp}}^+, d_{\text{mp}}^- = f_{nn}(\mathbf{h}_{\pi_{\text{mp}}^+}^{\neq t+}, \mathbf{h}_{G_i}^+)^2, f_{nn}(\mathbf{h}_{\pi_{\text{mp}}^-}^{\neq t-}, \mathbf{h}_{G_i}^-)^2$;

// Cost function Equation 12;
 $\text{Cost}_i = [d_{\text{node}}^+ - d_{\text{node}}^- + \alpha_1]_+ + [d_{\text{mp}}^+ - d_{\text{mp}}^- + \alpha_2]_+$;

end

end

$W_{i+1} := \text{miniBatchSGD}(W_i, \text{Cost}_i)$ // Weight updates;

250

251 4.3 Training Data Preparation

As introduced previously, G-HIN2VEC uses the negative sampling technique to generate training data. Therefore, data preparation is a crucial phase in our approach. As detailed in Algorithm 2 and illustrated in Figure 3, we train our model in mini-batches, sampled from \mathbb{G} , and then generate the positive context π_{mp}^+ from G_i with a predefined metapath $p^+ \sim_{\text{unif.}} \mathcal{P}$ following our custom sampling strategy.

$$Pr(v^i | v^{i-1}; p^+) = \begin{cases} \beta \frac{1}{|\mathcal{N}_{n_i}(v^{i-1})|} & e_i \in \mathcal{R}_G \text{ and } \tau_E(e_i) = r_i \\ (1 - \beta) \frac{1}{|\mathcal{N}_{\neq n_i}(v^{i-1})|} & e_i \in \mathcal{R}_G \text{ and } \tau_E(e_i) \neq r_i \\ 0 & e_i \notin \mathcal{R}_G; \text{ where } e_i = (v^i, v^{i-1}) \end{cases} \quad (15)$$

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

255
 256 As in Figure 3, our approach introduces noise to the original positive
 257 metapath instances π_{mp}^+ , creating negative contexts π_{mp}^- within the
 258 same graph rather than from different graphs. This is achieved by
 259 perturbing π_{mp}^+ with alternative metapaths p^- uniformly sampled
 260 from \mathcal{P} . The noise level $\lambda \in [0, 1]$ is set to 0.3; a process we denote as
 261 *MetapathSeqNoising*. After generating both positive and negative
 262 instances, we apply a series of linear transformations to derive the
 263 final triplet hidden representations to estimate the dissimilarity metrics.
 264
 265
 266
 267
 268
 269
 270
 271
 272

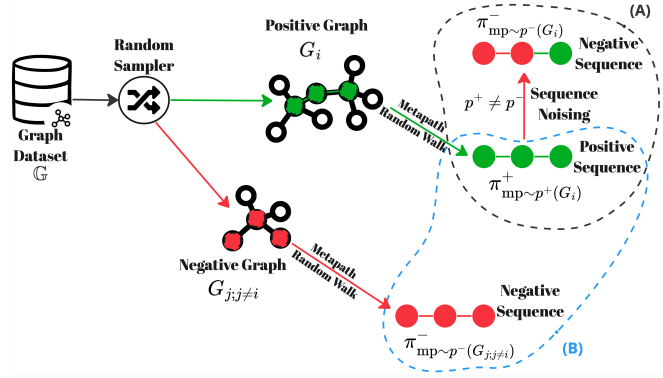


Fig. 3: G-HIN2VEC training data generation process, highlighting the **novel approach (A)** that deviates from **traditional methods (B)**. It showcases the creation of positive ● and negative ● samples in an unsupervised learning context.

273 Depending on the quantified metrics in our unsupervised framework, we optimize the
 274 global model weights by minimizing the double-triplet loss via back-propagation and
 275 gradient descent to learn meaningful graph-level embeddings for the heterogeneous
 276 graph dataset.

277 5 Experiments

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

283 5.1 Datasets and Tasks

284 We conducted our experiments on a dataset comprising records from 100K card-
285 holders and transactions from 2019 to 2021. The dataset’s schema includes the
286 transaction table: `cardholder_id` (as unique INT), `merchant_info` (as VARCHAR),
287 representing merchant code category and merchant name such as ‘5661-Shoe Stores’,
288 `merchant_location` (as VARCHAR), detailing the location like ‘Paris-France’, and
289 `transaction_date` (as DATE), indicating when the transaction occurred, for instance,
290 ‘2019-06-21’. This raw data was transformed into an ego-centric cardholder graph
291 dataset, as elaborated in Section 3.3. Furthermore, we used the cardholder table:
292 `cardholder_id` (as unique INT), `Gender` (as VARCHAR), `Age` (as FLOAT) and `Income`
293 (as FLOAT). All types are according to SQL data types.

294 The cardholder socio-demographic attributes used for supervised machine learning
295 experiments include:

- 296 • **Gender**, $Y_{Gender}^i \in \{0, 1\}, \forall i \in \{1, \dots, n\}$) as a binary attribute
indicating the i^{th} cardholder’s gender, where 0 represents one
gender and 1 represents another.
- 297 • **Income**, $Y_{Income}^i \in [0, 1], \forall i \in \{1, \dots, n\}$) as a normalized attribute
representing the i^{th} cardholder’s income.
- **Age**, $Y_{Age}^i \in [0, 1] \forall i \in \{1, \dots, n\}$ as a normalized attribute repre-
senting the i^{th} cardholder’s age.

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

303 5.2 Baselines and Experimental Settings

304 In this section, we outline the baseline methods used for comparison with our G-
305 HIN2VEC approach, focusing on homogeneous and heterogeneous graph embeddings,
306 as well as other graph-level representation techniques. These baselines are detailed in
307 Table 2 and grouped into three primary categories. Our G-HIN2VEC method is imple-
308 mented using the StellarGraph framework, a Python library for machine learning on
309 graphs. All graph embeddings for the cardholder were learned using G-HIN2VEC and
310 the baseline models. For G-HIN2VEC specifically, the experimental settings include a
311 dropout rate of 0.5, and the use of the SGD optimizer. We adjust the margin thresh-
312 olds α_1 and α_2 in our loss function, in Equation 12, to 1 and 0.5, respectively, with
313 a learning rate of 0.005. Additionally, weight decay is accompanied by an L2 penalty
314 set at 0.001 over 50 epochs.

315
316 After the pretraining phase on the graph dataset of cardholders, we benchmarked the
317 performance of the generated embeddings on a set of downstream tasks. Linear and
318 logistic regression were implemented on the graph embedding vector for regression and
319 classification tasks, respectively. The dataset was divided into training, validation, and

Category	Category Description	Method	Method Description
Vectorization-based	Uses node and graph features to represent graphs, focusing on node frequency, graph statistics, and structural properties such as centrality measures, graph density, and spectral properties.	N-grams ($N = 1, 2$)	Captures local graph features through individual and pairs of nodes frequency, emphasizing simpler patterns.
		Bag-of-Features (BoF)	Incorporates comprehensive graph features, including statistics on node degree, clustering coefficients, and triangles and squares sub-graph counts.
Kernel-based	Employing graph kernels, such as outlined in Equation 5 to map graphs into a vector space. This transformation facilitates the computation of a distance matrix for comparing graphs, as illustrated in Equation 7.	Graphlet kernel (GK) [23]	Represents graphs based on the frequency of small subgraphs (graphlets) throughout the graph dataset, ideal for capturing local topology.
		Shortest path kernel (SPK) [24]	Encodes each graph by the shortest path lengths between nodes within the graph dataset, reflecting global connectivity.
		Weisfeiler-Lehman framework (WLG) [33]	Refines graph representation through iterative relabeling based on neighborhood structures to capture structural changes over multiple scales, reflecting both local and global structural differences throughout the graph dataset.
		Deep GK, Deep SP, Deep WL [21]	Apply deep learning techniques with traditional graph kernels to generate graph-level embeddings. All variants capture different complex structural graph features in the graph dataset.
GNN-based	Uses deep learning architectures to generate node-level and graph-level embeddings, for node-level architectures an aggregation function is used to produce graph-level embeddings from node-level from node-level and edge-level embeddings, as in Equation 6.	Metapath2vec [16]	Uses guided random walks by metapaths in heterogeneous graphs to generate node embeddings. For graph-level features, the graph node embeddings are then aggregated using mean to produce graph-level embeddings.
		HIN2VEC [15]	Generate node and relation embeddings in heterogeneous networks by considering multiple types of nodes and relationships. For graph-level features, graph nodes and relationships are aggregated by average to form graph-level embeddings.
		Graph2Vec [20]	Treat the entire graph as an individual document in a corpus, employing a document embedding approach to learn graph embeddings. This method captures global graph properties and structural features.
		DiffPool [17]	Implement a differentiable graph pooling module that hierarchically aggregates nodes embeddings into clusters, forming a new, coarser graph at each layer, until final cluster as the graph-level representations.

Table 2: Detailed overview of graph-based methods categorized by their approach and core features, including graph-level and heterogeneity, with a focus on graph representations.

320 test sets based on the cardholder ID to ensure that all transactions related to a single
321 cardholder fall into the same group. The split ratio was 2:1:1 for training, validation,
322 and test sets, respectively. The performance results reported on Table 4 are the average
323 of 10-fold cross-validation test sets, with the statistical significance of each metric
324 validated by calculating the p-value. Performance for graph-level representation in the
325 classification task was evaluated using averaged accuracy, AUC, and F1 metrics, and
326 for regression tasks using R-squared (R2) and mean absolute error (MAE) metrics.

327 5.3 Empirical Validation

328 We performed empirical validation on the graph dataset following the experimental
329 settings described earlier. For non-graph-level representation models such as
330 HIN2Vec, we added a simple averaging layer on top of each node-level representa-
331 tion to generate graph representations. In heterogeneous guided random walk-based
332 GNNs, the definition of the metapath is required.

333
334 For this purpose, experts provided a set of metapaths to answer specific business
335 questions (BQ), as shown in Table 3. Furthermore, metapath-free baselines ignore
336 the predefined metapaths for both node- and graph-level representation methods,

337 and this will not be considered in both experimental analyses due to the original
 338 implementations.

339 5.4 Quantitative Analysis

Category	Method	Gender		Income		Age	
		Acc.	AUC	R2	MAE	R2	MAE
Vectorization based	N-gram (1 & 2-grams)	0.6133	0.601	0.1067	0.1531	0.1045	12.7074
	Bag of Features (BoF)	0.7295	0.722	0.1751*	0.1409*	0.2994	10.9955
Kernel based	Graphlet kernel [23] (GK)	0.5825	0.5647	0.0449	0.1618	0.0846	12.5174
	Shortest path kernel [24] (SPK)	0.5477	0.5002	0.0698	0.1597	0.0873	12.9465
	Weisfeiler-Lehman [33] (WL)	0.7001	0.6993	0.2030	0.1410	0.4301	9.8510
Deep Kernel based [21]	Deep GK	0.5399	0.5185	0.0472	0.1469	0.0842	11.4422
	Deep SP	0.5566	0.5137	0.0798	0.1629	0.0922	13.1960*
	Deep WL	0.7032*	0.7292**	0.2255	0.1381*	0.4717	9.5978**
GNN based	Metapath2vec [‡] [16]	0.6102	0.5586	0.1105**	0.1489	0.1349	12.3272
	Graph2Vec ^{†§} [20]	0.7220	0.7331	0.2394*	0.1354	0.5384	9.4021
	HIN2VEC [‡] [15]	0.7809	0.7906	0.2318	0.1399*	0.6385*	6.6019
	DiffPool ^{†§} [17]	0.8047*	0.8194	0.3088**	0.1212**	0.7108*	5.9277*
	G-HIN2VEC ^{‡§}	0.8244*	0.8311*	0.3310*	0.1137**	0.6843**	6.3157*

Table 4: Performance of various graph-based methods in predicting cardholder attributes. Superscripts denote model compatibility: [‡] for heterogeneous graphs, [†] for homogeneous graphs, ^{||} for node-level, and [§] for graph-level downstream tasks. Statistical significance is denoted as: ** $p < 0.01$, * $p < 0.05$.

340 5.4.1 Regression tasks.

341 In regression tasks targeting age
 342 and income prediction, our evalua-
 343 tion reveals that vectorization and
 344 graph kernel-based methods strug-
 345 gle to effectively model the complex
 346 relationships inherent in heteroge-
 347 neous graph data, resulting in infer-
 348 rior performance. In contrast, GNN-
 349 based models show enhanced out-
 350 comes due to their advanced archi-
 351 tectures, which are better suited
 352 to capture the complex interactions
 353 within these graphs. Notably, our G-
 354 HIN2VEC model demonstrates sig-
 355 nificant improvements over the strong baseline, DiffPool, with a 7.18% increase in
 356 R-squared and a 6.19% decrease in mean absolute error for income prediction, despite
 357 its non-Gaussian distribution. For age prediction, which follows a Gaussian distri-
 358 bution, G-HIN2VEC matches the performance of DiffPool. These results affirm the
 359 robustness of G-HIN2VEC in generating effective graph-level embeddings for regres-
 360 sion tasks, highlighting its potential for broader application in complex graph-based
 361 analysis.

Table 3: Set of business questions as metapaths by financial experts, where ● Merchant, ● Location, and ● Time stand for node types.

BQ 1:	What spending habit does the cardholder have?
Metapath 1	● _M :Next: ● _M
BQ 2:	Where does the cardholder use the credit card?
Metapath 2	● _M :Located_in: ● _L :Located_in: ● _M
BQ 3:	When does the cardholder use the credit card?
Metapath 3	● _M :Available_at: ● _T :Available_at: ● _M
BQ 4:	When and where is the credit card used?
Metapath 4	● _M :Available_at: ● _L :Located_at: ● _T :Available_at: ● _M

362 5.4.2 Classification task.

363 The gender classification performance of various methods is comprehensively detailed
364 in Table 4. Among the benchmarked methods, vectorization methods are observed to
365 outperform graph kernel-based methods, showcasing their better suitability for clas-
366 sification tasks involving categorical data. In contrast, among the baselines, DiffPool
367 stands out as the most effective model. It highlights the benefits of utilizing sophis-
368 ticated aggregation techniques rather than averaging at the top of the node-level,
369 thus enhancing the quality of graph-level embeddings. In the specific case of our G-
370 HIN2VEC model, significant improvements in gender classification are evident, with
371 increases of 2.44% in accuracy, 1.85% in F1 score, and 1.43% in AUC compared to
372 DiffPool. These statistically significant results demonstrate G-HIN2VEC’s superior
373 capability in leveraging complex graph-level embeddings for classification tasks.
374 Additionally, the analysis of the graph
375 structures within our dataset, particu-
376 larly their size distribution and nega-
377 tive assortativity coefficients as shown
378 in Figure 4, supports these outcomes.
379 This structural characteristic suggests
380 that G-HIN2VEC not only effectively
381 handles disassortative graphs but also
382 benefits from adjustments such as the
383 incorporation of a teleportation term, β ,
384 in our sampling strategy (see Equation
385 15), which enhances model robustness
386 and confirms the quantitative findings
387 of improved performance. Furthermore,
388 a primary limitation of GNN models, as highlighted in [35], involves their sensitiv-
389 ity to assortativity during node-level tasks, which can extend to graph-level tasks,
390 potentially impacting performance; our model’s design addresses and mitigates this
391 limitation effectively.

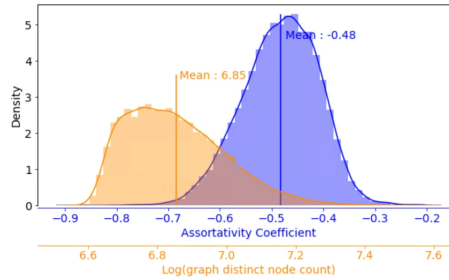


Fig. 4: The probability distribution of (a) the log graph sizes and (b) the assortativity coefficient of graphs in the dataset.

392 5.5 Qualitative analysis

393 In this section, we investigated the cluster assignment of cardholder behavior through
394 the graph-level embeddings. Figure 5 presents a t-SNE visualization of cardholder
395 embeddings, where each color signifies a distinct cluster, corresponding to a unique
396 behavioral pattern in credit card usage. Our assumption is that cardholders who are
397 semantically similar in using credit cards tend to be embedded in close proximity, as
398 shown in [20] and [16] for graph- and node-level representations, respectively.

399
400 The clustering of embeddings into 16 distinct groups, as determined by the elbow
401 method using k-means, reveals a granular view of cardholder behaviors, with each
402 cluster capturing a particular lifestyle preference that aligns with previous findings in
403 credit card usage research [1]. In post-clustering analysis, we engaged domain experts
404 to identify cluster identities, extracting and examining metapath instances that were

405 sampled during training. This analysis aimed to quantify the occurrence of directed
 406 sequences within each cluster, thus providing a narrative for the lifestyle categories
 407 annotated in Figure 5. Thus, two cardholders are semantically similar if their credit
 408 card usage behavior is similar with respect to a set of predefined metapaths in Table
 409 3.

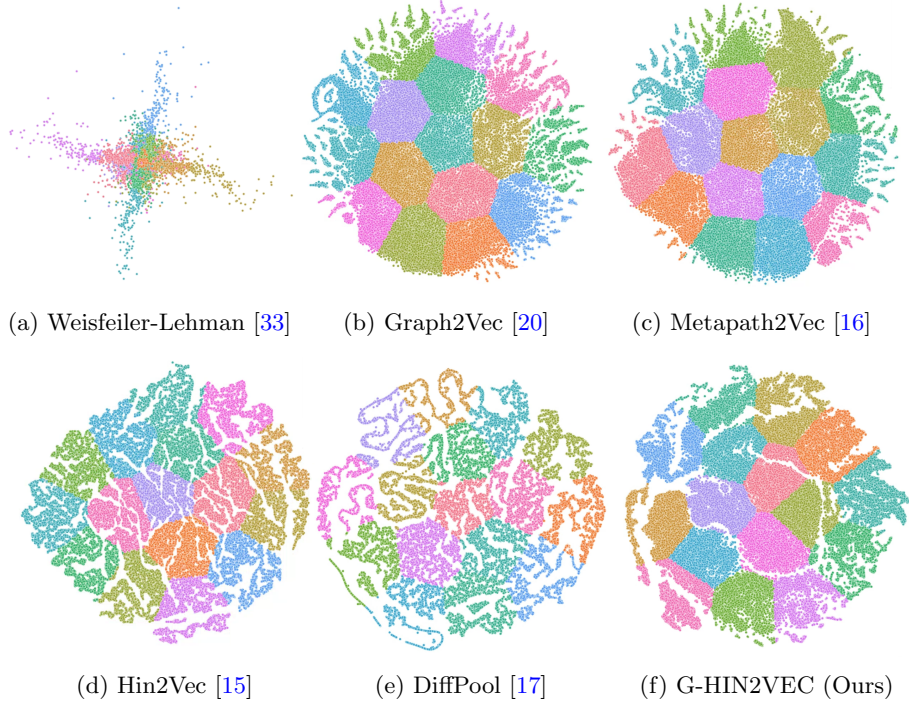


Fig. 5: Cardholder embeddings t-SNE visualization of 16 identified clusters using different embedding algorithms. Each color represents a distinct cluster; ● Grocery Shoppers, ● Commuters ● Young Workers ● High-Tech ● Elderly ● Restaurant-goers ● Drivers ● Musicophiles ● Bookworm/Bibliophiles ● Business ● FastFood-goers ● Householders ● Cash-Only ● Gamblers ● Peripatetic/Travelers ● Epicureans.

410 The t-SNE visualizations highlight how well different graph-level embedding meth-
 411 ods group similar cardholder behaviors. The Weisfeiler-Lehman method results in
 412 overlapping clusters, suggesting that it may not be as effective in identifying the finer
 413 details of the graph-level features. Graph2Vec and Metapath2Vec improve upon this
 414 with more defined clusters, reflecting their stronger ability to map out both the indi-
 415 vidual elements and graph-level structures. HIN2VEC goes a step further, creating
 416 even more distinct clusters that show its capability to capture the variety within
 417 the graph dataset, and the DiffPool method stands out with its highly distinct and
 418 compact clusters, which means that it is particularly good at understanding complex
 419 patterns. Our G-HIN2VEC method generates embeddings that are more distinct than

420 Metapath2Vec but not as tightly clustered as HIN2VEC and DiffPool, suggesting that
421 it provides a detailed, yet comprehensive overview of cardholder behavior. This aligns
422 with the solid results we see from our quantitative analysis, where our method shows
423 a strong ability to accurately group cardholder behaviors.

424 **6 Conclusion And Future Work**

425 In this paper, we introduce G-HIN2VEC, a novel approach to generate heteroge-
426 neous graph-level embeddings. Our method utilizes a double-triplet loss and an
427 unsupervised learning framework that incorporates negative sampling, eliminating
428 the need for graph-to-graph matching. This allows for more efficient learning without
429 relying on negative samples from the entire graph dataset. We used a real-world
430 financial dataset, modeling it as ego-centric graphs for cardholders to benchmark
431 various graph-level representation models. Our results show competitive performance
432 of G-HIN2VEC with state-of-the-art models, with notable improvements in gender
433 classification accuracy of 2.45% and income prediction R-squared (R²) of 7.19%.
434 Furthermore, for age prediction, we achieved a 6.55% increase in mean absolute error
435 (MAE) compared to the strong baseline, DiffPool.

436
437 Looking ahead, we plan to enhance G-HIN2VEC by integrating dynamic metapath
438 selection mechanisms to address node and edge type imbalances in Heterogeneous
439 Information Networks (HINs). We also aim to extend the application of G-HIN2VEC
440 to a wider range of graph-level downstream tasks. Exploring innovative negative
441 sampling techniques, such as sequence noising and nonsampling approaches, will
442 be another focus to further improve graph-level representation learning within an
443 unsupervised framework. A current limitation of our methodology is its reliance on
444 a transductive learning model, which may not perform well with unseen data. To
445 overcome this, our future work will focus on an inductive model that can generalize to
446 new, unseen graph instances, enhancing the applicability and utility of our approach
447 in the field of graph representation learning.

448 **Acknowledgments**

449 We would like to acknowledge the invaluable knowledge exchange facilitated by the
450 Data Analysis Lab team and our industry partners' experts. Partial funding for this
451 work was provided by the Luxembourg National Research Fund (FNR) under grant
452 number 15829274, supplemented by ANR-20-CE39-0008.

453 **Data Availability**

454 Due to privacy considerations and adherence to the General Data Protection Regula-
455 tion in the European Union (GDPR) as defined in Article 4, the data supporting this
456 study are not publicly available.

457 **Declarations**

458 **Conflict of Interest**

459 The authors declare that they have no conflict of interest.

460 **Ethical Approval**

461 The authors approve that the research presented in this article is conducted following
462 the principles of ethical and professional conduct.

463 **Consent to Participate**

464 Not applicable.

465 **Consent for Publication**

466 Not applicable.

467 **References**

- 468 [1] Di Clemente, R., Luengo-Oroz, M., Travizano, M., Xu, S., Vaitla, B., González,
469 M.C.: Sequences of purchases in credit card data reveal lifestyles in urban
470 populations. *Nature communications* **9**(1), 3330 (2018)
- 471 [2] Ren, Y., Zhu, H., Zhang, J., Dai, P., Bo, L.: Ensemfdet: An ensemble approach
472 to fraud detection based on bipartite graph. In: 2021 IEEE 37th International
473 Conference on Data Engineering (ICDE), pp. 2039–2044 (2021). IEEE
- 474 [3] Khazane, A., Rider, J., Serpe, M., Gogoglou, A., Hines, K., Bruss, C.B., Serpe,
475 R.: Deeptrax: Embedding graphs of financial transactions. In: 2019 18th IEEE
476 International Conference On Machine Learning And Applications (ICMLA), pp.
477 126–133 (2019). IEEE
- 478 [4] Wang, R., Zhuang, Z., Tao, H., Paszke, W., Stojanovic, V.: Q-learning based fault
479 estimation and fault tolerant iterative learning control for mimo systems. *ISA*
480 *transactions* **142**, 123–135 (2023)
- 481 [5] Song, X., Wu, N., Song, S., Zhang, Y., Stojanovic, V.: Bipartite synchroniza-
482 tion for cooperative-competitive neural networks with reaction–diffusion terms
483 via dual event-triggered mechanism. *Neurocomputing* **550**, 126498 (2023)
- 484 [6] Song, X., Wu, N., Song, S., Stojanovic, V.: Switching-like event-triggered state
485 estimation for reaction–diffusion neural networks against dos attacks. *Neural*
486 *Processing Letters* **55**(7), 8997–9018 (2023)
- 487 [7] Vishwanathan, S.V.N., Schraudolph, N.N., Kondor, R., Borgwardt, K.M.: Graph
488 kernels. *The Journal of Machine Learning Research* **11**, 1201–1242 (2010)

- 489 [8] Goyal, P., Ferrara, E.: Graph embedding techniques, applications, and perfor-
490 mance: A survey. *Knowledge-Based Systems* **151**, 78–94 (2018)
- 491 [9] Wang, X., Bo, D., Shi, C., Fan, S., Ye, Y., Philip, S.Y.: A survey on hetero-
492 geneous graph embedding: methods, techniques, applications and sources. *IEEE*
493 *Transactions on Big Data* **9**(2), 415–436 (2022)
- 494 [10] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive
495 survey on graph neural networks. *IEEE transactions on neural networks and*
496 *learning systems* **32**(1), 4–24 (2020)
- 497 [11] Cai, H., Zheng, V.W., Chang, K.C.-C.: A comprehensive survey of graph embed-
498 ding: Problems, techniques, and applications. *IEEE transactions on knowledge*
499 *and data engineering* **30**(9), 1616–1637 (2018)
- 500 [12] Ma, G., Ahmed, N.K., Willke, T.L., Yu, P.S.: Deep graph similarity learning: A
501 survey. *Data Mining and Knowledge Discovery* **35**, 688–725 (2021)
- 502 [13] Liu, Z., Chen, C., Yang, X., Zhou, J., Li, X., Song, L.: Heterogeneous graph
503 neural networks for malicious account detection. In: *Proceedings of the 27th ACM*
504 *International Conference on Information and Knowledge Management*, pp. 2077–
505 2085 (2018)
- 506 [14] Sun, L., He, L., Huang, Z., Cao, B., Xia, C., Wei, X., Philip, S.Y.: Joint embedding
507 of meta-path and meta-graph for heterogeneous information networks. In: *2018*
508 *IEEE International Conference on Big Knowledge (ICBK)*, pp. 131–138 (2018).
509 IEEE
- 510 [15] Fu, T.-y., Lee, W.-C., Lei, Z.: Hin2vec: Explore meta-paths in heterogeneous
511 information networks for representation learning. In: *Proceedings of the 2017*
512 *ACM on Conference on Information and Knowledge Management*, pp. 1797–1806
513 (2017)
- 514 [16] Dong, Y., Chawla, N.V., Swami, A.: metapath2vec: Scalable representation learn-
515 ing for heterogeneous networks. In: *Proceedings of the 23rd ACM SIGKDD*
516 *International Conference on Knowledge Discovery and Data Mining*, pp. 135–144
517 (2017)
- 518 [17] Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W., Leskovec, J.: Hierarchi-
519 cal graph representation learning with differentiable pooling. *Advances in neural*
520 *information processing systems* **31** (2018)
- 521 [18] Wang, Y., Duan, Z., Liao, B., Wu, F., Zhuang, Y.: Heterogeneous attributed
522 network embedding with graph convolutional networks. In: *Proceedings of the*
523 *AAAI Conference on Artificial Intelligence*, vol. 33, pp. 10061–10062 (2019)

- 524 [19] Yang, C., Xiao, Y., Zhang, Y., Sun, Y., Han, J.: Heterogeneous network rep-
525 resentation learning: A unified framework with survey and benchmark. *IEEE*
526 *Transactions on Knowledge and Data Engineering* **34**(10), 4854–4873 (2020)
- 527 [20] Narayanan, A., Chandramohan, M., Venkatesan, R., Chen, L., Liu, Y.,
528 Jaiswal, S.: graph2vec: Learning distributed representations of graphs. *ArXiv*
529 **abs/1707.05005** (2017)
- 530 [21] Yanardag, P., Vishwanathan, S.: Deep graph kernels. In: *Proceedings of the*
531 *21th ACM SIGKDD International Conference on Knowledge Discovery and Data*
532 *Mining*, pp. 1365–1374 (2015)
- 533 [22] Bai, Y., Ding, H., Qiao, Y., Marinovic, A., Gu, K., Chen, T., Sun, Y., Wang,
534 W.: Unsupervised inductive graph-level representation learning via graph-graph
535 proximity, 1988–1994 (2019)
- 536 [23] Borgwardt, K.M., Kriegel, H.-P.: Shortest-path kernels on graphs. In: *Fifth IEEE*
537 *International Conference on Data Mining (ICDM'05)*, p. 8 (2005). *IEEE*
- 538 [24] Shervashidze, N., Vishwanathan, S., Petri, T., Mehlhorn, K., Borgwardt, K.: Effi-
539 cient graphlet kernels for large graph comparison. In: *Artificial Intelligence and*
540 *Statistics*, pp. 488–495 (2009). *PMLR*
- 541 [25] Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural
542 networks? In: *International Conference on Learning Representations* (2018)
- 543 [26] Rao, S.X., Zhang, S., Han, Z., Zhang, Z., Min, W., Chen, Z., Shan, Y., Zhao,
544 Y., Zhang, C.: xfraud: explainable fraud transaction detection. *arXiv preprint*
545 *arXiv:2011.12193* (2020)
- 546 [27] Zheng, Y., Lee, V.C., Wu, Z., Pan, S.: Heterogeneous graph attention network
547 for small and medium-sized enterprises bankruptcy prediction. In: *Pacific-Asia*
548 *Conference on Knowledge Discovery and Data Mining*, pp. 140–151 (2021).
549 *Springer*
- 550 [28] Liu, Z., Wang, D., Yu, Q., Zhang, Z., Shen, Y., Ma, J., Zhong, W., Gu, J., Zhou, J.,
551 Yang, S., *et al.*: Graph representation learning for merchant incentive optimization
552 in mobile payment marketing. In: *Proceedings of the 28th ACM International*
553 *Conference on Information and Knowledge Management*, pp. 2577–2584 (2019)
- 554 [29] Wang, J., Zhang, S., Xiao, Y., Song, R.: A review on graph neural network
555 methods in financial applications. *Journal of Data Science* **20**(2), 111–134 (2022)
- 556 [30] Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In:
557 *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge*
558 *Discovery and Data Mining*, pp. 855–864 (2016)

- 559 [31] Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social represen-
560 tations. In: Proceedings of the 20th ACM SIGKDD International Conference on
561 Knowledge Discovery and Data Mining, pp. 701–710 (2014)
- 562 [32] Dyreson, C.E., Evans, W.S., Lin, H., Snodgrass, R.T.: Efficiently supporting
563 temporal granularities. *IEEE Transactions on Knowledge and Data Engineering*
564 **12**(4), 568–587 (2000)
- 565 [33] Shervashidze, N., Schweitzer, P., Van Leeuwen, E.J., Mehlhorn, K., Borgwardt,
566 K.M.: Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*
567 **12**(9) (2011)
- 568 [34] Wang, F., Zuo, W., Lin, L., Zhang, D., Zhang, L.: Joint learning of single-image
569 and cross-image representations for person re-identification. In: Proceedings of the
570 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1288–1296
571 (2016)
- 572 [35] Suresh, S., Budde, V., Neville, J., Li, P., Ma, J.: Breaking the limit of graph neural
573 networks by improving the assortativity of graphs with local mixing patterns. In:
574 Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery &
575 Data Mining, pp. 1541–1551 (2021)