LoRE: Robust and Adaptive Graph Embeddings via Local Self-Reconstruction Mechanisms

Anonymous authors
Paper under double-blind review

Abstract

Graph embeddings aim to project nodes into numeric vector spaces while preserving structural and semantic properties for downstream tasks such as community detection, node classification, and link prediction. However, existing embedding methods distort local geometry through negative sampling, fail to enforce semantic consistency, and require expensive retraining when graphs evolve. Therefore, we introduce LoRE, a versatile graph embedding framework based on attention-driven self-reconstruction mechanisms and a perspective-preserving training procedure. Built on a generalized formulation, LoRE can be applied to a wide range of graph types, from undirected graphs to relational knowledge graphs and even attributed node sets without inherent topologies. It enforces identical embeddings for structurally equivalent nodes, respects local context during training, and reduces the likelihood of violations of the open-world assumption. Unlike traditional methods, LoRE supports efficient on-the-fly adaptation: embeddings can be updated in real time as graphs change, without full retraining. Its reconstruction mechanism acts as a self-supervised training signal that improves embedding robustness, yielding superior performance compared to existing approaches. Extensive experiments demonstrate that LoRE consistently matches or outperforms baseline results while maintaining stability under dynamic conditions. Qualitative analyses further show that LoRE produces more separable and compact clusters in embedding spaces. Together, these results underscore its enhanced generalizability and practical value in real-world graph embedding applications.

1 Introduction

Graphs provide a natural framework for representing data with inherent relational structure. Given a set of nodes, they capture relationships between them as edges and may include additional metadata, such as edge directions, weights, or labels. Such representations are central to a wide range of domains, including biomedicine, finance, and manufacturing (Zhou et al., 2020). Knowledge graphs (KGs) further extend this paradigm by supporting logical inference through axioms encoded in ontologies, enabling the integration of statistical and symbolic reasoning (Hogan et al., 2021). To make graph-structured data usable for downstream tasks, graph embedding (GE) methods map nodes into continuous vector spaces (Wang et al., 2017). For instance, this transformation enables machine learning tasks such as node classification, link prediction, and graph-based recommendation (Nickel et al., 2016). Existing GE methods are usually self-supervised and are commonly organized into four main categories: factorization-based approaches (Nickel et al., 2011; Ou et al., 2016), random-walk-based techniques (Ristoski & Paulheim, 2016; Grover & Leskovec, 2016), translational distance models (Bordes et al., 2013; Lin et al., 2015), and graph neural networks (GNNs) (Kipf & Welling, 2017; Veličković et al., 2018). Despite their successes, several practical requirements remain unmet. In this work, we identify four key properties essential for robust and adaptive embeddings in real-world applications, reflecting challenges such as noise, dynamics, and consistency:

- P1 **Neighborhood-Invariance:** Structurally indistinguishable nodes yield identical embeddings.
- P2 **OWA Obedience:** Training procedures avoid violating the open world assumption (OWA).
- P3 **Locality-Awareness:** Local relationships must be preserved during the embedding process.
- P4 **Graph Dynamics:** Embeddings must adapt to graph updates in real time without retraining.

These properties constitute critical requirements for practical implementations, yet, as we show in Section 3, current GE approaches do not consistently meet them. To overcome these challenges, we propose the LORE approach (LOcally Reconstructed Embeddings). It reconstructs randomly initialized base embeddings via attention mechanisms, using only the embeddings of immediate neighbors and their relation types as input. The resulting reconstructions are used as the final embeddings, enforcing neighborhood-invariance **(P1)** by ensuring that structurally identical nodes yield identical representations. Unlike traditional edge-level negative sampling, LORE draws node-level negatives for its multi-edge reconstructions of positive target node embeddings, thereby reducing the risk of violating the OWA **(P2)**. When combined with margin-ranking loss (MRL) using fixed-perspective distances, embeddings of plausibly similar nodes may be pushed apart. To avoid such distortions, LORE employs a perspective-preserving angular loss that respects relative similarities of positive and negative samples, maintaining local relationships while enforcing necessary separation during training **(P3)**. Finally, since embeddings are reconstructed from local attributes, LORE naturally adapts to graph updates through updated forward passes **(P4)**. Extensive evaluation on benchmark tasks confirms that LORE not only outperforms existing baselines but also demonstrates strong dynamic capabilities, underscoring its potential as a robust and versatile solution for real-world graphs.

2 PRELIMINARIES & DEFINITIONS

In this section, we introduce the principles and definitions of graphs and their embeddings, treating embeddings as continuous vector representations of nodes. To unify the variety of graph types and embedding methods, we formalize LORE as a generalized framework that can also encompass unstructured yet attribute-enriched node sets and their embeddings.

2.1 GRAPHS

Graphs provide a flexible formalism for representing entities as nodes and their relationships as edges, optionally enriched with metadata such as labels or weights. This abstraction underlies a broad range of real-world systems, including social and citation networks, molecular structures, and knowledge bases. In particular, KGs have become a widely adopted paradigm for structuring and integrating heterogeneous information. Regardless of their complexity, every graph induces an underlying simple graph that retains only its basic connectivity, the simplest form of graph structure.

Definition 1. *Given a set of nodes $\mathcal{V}$ and edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, a simple graph (SG) is defined as*

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}).$$

It is called undirected if the edges $(u, v) \in \mathcal{E}$ are unordered pairs, and directed otherwise. Each directed SG induces an undirected SG by symmetrizing its edges, that is, $(u, v) \in \mathcal{E} \Rightarrow (v, u) \in \mathcal{E}$.

SGs thus provide a minimal representation of graphs via pairwise connections, directed or undirected. Many real-world graphs, however, enrich edges with labels to encode additional semantics.

Definition 2. *Given a node set $\mathcal{V}$, a set of labels $\mathcal{L}$, and labeled edges $\mathcal{E}_\mathcal{L} \subseteq \mathcal{V} \times \mathcal{V} \times \mathcal{L}$, a labeled graph (LG) is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E}_\mathcal{L})$. Directedness follows analogously to SGs. An LG is undirected if and only if $(u, v, l) \in \mathcal{E}_\mathcal{L}$ implies $(v, u, l) \in \mathcal{E}_\mathcal{L}$, and directed otherwise.*

Each LG thus induces an SG by collapsing its edges along the labels to prevent duplicates. Among LGs, knowledge graphs form an important subclass where labels correspond to semantic relations.

Definition 3. *Let $\mathcal{R} \equiv \mathcal{L}$ be a finite set of relations. A directed LG $\mathcal{G} = (\mathcal{V}, \mathcal{E}_\mathcal{R})$ is called knowledge graph (KG), and an edge $(h, t, r) \in \mathcal{E}_\mathcal{R}$ denotes a relation of type $r \in \mathcal{R}$ from head h to tail t.*

Most real-world KGs additionally encode literals, i.e., non-contextual values such as numbers, strings, or timestamps. To account for this, we can represent the node set as the union $\mathcal{V} = \mathcal{V}_c \cup \mathcal{V}_l$ of contextual and literal-valued nodes, with edges $(h, t, r) \in \mathcal{E}_\mathcal{R}$ involving literals only as tails $t \in \mathcal{V}_l$. This way, literals can be regarded as node-level attributes, which we do not explicitly exploit in this work but include for generality. Accordingly, we introduce the notion of attributed node sets.

Definition 4. *Let $\mathcal{V}$ be a set of nodes and $\mathcal{A}$ a non-empty set of attributes. An attributed node set is a subset $\mathcal{V}_\mathcal{A} \subseteq \mathcal{V} \times \mathcal{A}$ equipped with the corresponding characteristic mapping*

$$c(\mathcal{V}_\mathcal{A}, \cdot) : \mathcal{V} \to 2^\mathcal{A}, \qquad c(\mathcal{V}_\mathcal{A}, v) = \{\, a \in \mathcal{A} \mid (v, a) \in \mathcal{V}_\mathcal{A} \,\},$$

where $2^\mathcal{A} = \{\, X \subseteq \mathcal{A} \,\}$ denotes the power set of $\mathcal{A}$ and $c(\mathcal{V}_\mathcal{A}, v)$ returns the attributes of $v \in \mathcal{V}$.

This definition formalizes how attributes are associated with nodes without imposing any specific topology. Although LoRE can incorporate such attributes once they are suitably projected into an embedding space, handling literal values such as continuous numbers or free text is non-trivial when learning graph-based embeddings (Gesese et al., 2021). Following common practice in GE methods, we therefore omit these attributes in our experiments, except when the attribute labels form finite sets that can be transformed into contextual node structures. Finally, we define graphs as attributed node sets whose attributes represent adjacencies and can be augmented with metadata.

Definition 5. *Let $\mathcal{V}$ be a set of nodes, let $n \in \mathbb{N}_0$, and let $\mathcal{M}_1, \ldots, \mathcal{M}_n$ be non-empty attribute sets. An attributed node set $\mathcal{V}_\mathcal{A}$ induces a (contextual) graph $\mathcal{G} = (\mathcal{V}, \mathcal{V}_\mathcal{A})$ if and only if*

$$\mathcal{A} = \mathcal{V} \times \mathcal{M}, \qquad \mathcal{M} = \prod_{i=1}^{n} \mathcal{M}_i,$$

where ordered pairs $(v, a) \in \mathcal{V}_\mathcal{A}$ link each node $v \in \mathcal{V}$ to its topological attributes $a \in \mathcal{A}$. Under this convention, directions must be explicitly encoded within edge metadata $m = (m_1, ..., m_n) \in \mathcal{M}$. Otherwise, incoming and outgoing edges are indistinguishable, i.e., the graph would be undirected.

Throughout this paper, we exclude self-edges, i.e., $(v, v, m) \notin \mathcal{V}_\mathcal{A}$ for any $v \in \mathcal{V}$ and $m \in \mathcal{M}$. Although they could be included in principle, they would be masked by LoRE's independent self-reconstruction mechanism and thus provide no additional benefit for our purposes. In this view, Definition 5 treats adjacent edges as node attributes, where each attribute corresponds to another node linked to it and may carry additional edge metadata $m \in \mathcal{M}$ such as relation types, weights, or timestamps, thereby unifying common graph structures. For example, an undirected SG is obtained by choosing $\mathcal{A} = \mathcal{V}$ so that $(u, v) \in \mathcal{V}_\mathcal{A}$ represents an edge between u and v, and $c_{\mathcal{V}_\mathcal{A}}(v)$ corresponds to the neighbors of v. To represent directedness, we include orientation in the attributes using $\mathcal{O} = \{o_{in}, o_{out}\}$ and choose $\mathcal{A} = \mathcal{V} \times \mathcal{O}$ so that $(h, t, o_{out}) \in \mathcal{V}_\mathcal{A}$ defines a directed edge from head h to tail t, with $(h, t, o_{out}) \in \mathcal{V}_\mathcal{A}$ implying $(t, h, o_{in}) \in \mathcal{V}_\mathcal{A}$. LGs can be defined analogously using a label set. In particular, KGs are represented by choosing $\mathcal{A} = \mathcal{V} \times \mathcal{O} \times \mathcal{R}$ with relation labels $\mathcal{R}$ so that $(h, t, o_{out}, r) \in \mathcal{V}_\mathcal{A}$ represents a directed edge of relation type $r \in \mathcal{R}$ from h to t.

2.2 Graph Embeddings

Analogous to the introduction of graphs, we define their embeddings from a generalized perspective. Building on the concept of attributed node sets, we formalize them as follows.

Definition 6. *Let $\mathcal{V}$ be a set of nodes. A mapping $\phi : \mathcal{V} \to \mathbb{R}^d$ is called a node embedding of dimension $d \in \mathbb{N}$, and $\phi(v)$ denotes the embedding of $v \in \mathcal{V}$. If $\mathcal{V}$ carries attributes specified by an attributed node set $\mathcal{V}_\mathcal{A}$ and there exists a projection $\pi : 2^\mathcal{A} \to \mathbb{R}^d$ such that*

$$\phi(v) = \pi(c(\mathcal{V}_\mathcal{A}, v)),$$

then ϕ is called an independent node embedding with respect to the attributes specified by $\mathcal{V}_\mathcal{A}$. When $\mathcal{V}_\mathcal{A}$ identifies a graph $\mathcal{G}_\mathcal{A}$, ϕ is also referred to as an (independent) graph embedding (GE).

Some works use the term graph embedding to denote a single vector for an entire graph. However, in this paper we refer to the collection of node embeddings within a graph. Moreover, we focus on real-valued embeddings in $\mathbb{R}^d$, though the LoRE approach naturally extends to other normed vector spaces, such as complex-valued embeddings (Trouillon et al., 2016; Sun et al., 2019). In standard practice, node embeddings are initialized randomly per node and optimized afterwards, introducing noise that may not be fully corrected during training. To ensure consistency and specify the key property of neighborhood-invariance (P1), we formalize the requirement that nodes with identical attributes (e.g., identical neighborhoods) map to identical embeddings.

Definition 7. *Let $\mathcal{V}_\mathcal{A}$ be an attributed node set with the node embedding $\phi : \mathcal{V} \to \mathbb{R}^k$. We call the embedding attribute-invariant if and only if identical attributes yield identical embeddings:*

$$c(\mathcal{V}_\mathcal{A}, u) = c(\mathcal{V}_\mathcal{A}, v) \quad \Rightarrow \quad \phi(u) = \phi(v).$$

This captures the idea that structurally indistinguishable nodes cannot be distinguished by the embedding. Moreover, attribute-invariant GEs are also referred to as neighborhood-invariant.

Independent GEs are therefore neighborhood-invariant and fulfill key property P1. Since LoRE's self-reconstructions rely solely on neighborhood attributes, they are independent embeddings themselves, a property that LoRE leverages to enable generalization and support dynamic capabilities.

3 Related Work

Real-world graphs are often incomplete, dynamic, and sparse (Xu, 2021; Krause et al., 2022), implying requirements for practical implementations of GEs, from which we formalize four key properties P1–P4. First, distinct nodes can share identical neighborhoods, especially in sparse graphs. Neighborhood-invariance **(P1)**, as per Definition 7, ensures that such nodes are treated consistently, which is essential for fair and stable predictions (Zemel et al., 2013), particularly in high-risk domains. Next, negative sampling is a common strategy for defining the training loss in self-supervised GE approaches, where edges are corrupted to create negative training targets (Kamigaito & Hayashi, 2022), typically combined with MRL to penalize negative facts. However, under OWA, the absence of an edge does not imply its incorrectness. Using corrupted facts as training objectives can therefore violate OWA in graphs, such as KGs (Hogan et al., 2021), by penalizing true but unobserved facts. OWA obedience **(P2)** requires that the training procedure minimizes the likelihood of such violations. Moreover, while computationally efficient, negative sampling can distort the local geometry of the embedding space if negative target nodes that are actually similar to the positives are pushed apart. Locality-awareness **(P3)** preserves similarities within the embedding space during training: negative target embeddings should be treated differently depending on their proximity to positive ones to avoid unnecessary distortion of local geometry. Finally, adaptability of embeddings to graph dynamics in real-time without retraining **(P4)** is essential when graphs contain noise and evolve over time, as is often the case for real-world graphs (Yang et al., 2024). In conclusion, P1–P3 define conditions under which an embedding can be considered robust, whereas P4 describes dynamic capabilities. We now review four major categories of GE methods with respect to these properties.

Factorization-based approaches approximate adjacency matrices $\mathcal{X}$ of graphs using low-rank decompositions (Cui et al., 2017). For instance, RESCAL (Nickel et al., 2011) factorizes $\mathcal{X} \approx ARA^\top$ into node embeddings A via interaction matrices R. Since A is randomly initialized and optimization noise can cause embeddings of structurally equivalent nodes to diverge, such methods do not fulfill P1. By treating unseen facts as zeros during optimization, they also violate P2, and they do not explicitly convey P3 since the objective is global reconstruction of the full adjacency matrices rather than preservation of local patterns. Furthermore, updates require retraining, violating P4.

Random-walk-based techniques such as Node2Vec (Grover & Leskovec, 2016) and RDF2Vec (Ristoski & Paulheim, 2016) generate sequences of nodes (and labels for labeled graphs) via random walks and learn embeddings using word2vec-style objectives. Together with random initialization of embeddings, they therefore do not fulfill P1, and negative sampling of nodes or labels that have not co-occurred in a walk violates P2. These methods capture co-occurrence patterns via context windows, which indirectly preserve local neighborhoods and similarities (P3) (Portisch & Paulheim, 2024; Khoshraftar & An, 2024). In contrast, adapting to graph updates requires both recomputing random walks and retraining (Hahn & Paulheim, 2024), so P4 is not satisfied.

Translational models are particularly designed for KGs to represent relations as translations in vector spaces. They aim to predict a tail embedding $\phi(t)$ from a head embedding $\phi(h)$ and a relation $r \in \mathcal{R}$ for a directed triple from head h to tail t via relation r. Several approaches exist (Wang et al., 2017), such as TransE (Bordes et al., 2013), TransR (Lin et al., 2015), and TransD (Ji et al., 2015). For instance, TransE assumes $\phi(h) + \rho(r) \approx \phi(t)$ with auxiliary relation embeddings $\rho(r)$, effectively approximating $\phi(t)$. Random initialization and training noise cause these models to fail P1. Corrupted triples are treated as false, violating P2. MRL with fixed-perspective distances, such as $\|\cdot\|_2$, are used with negative sampling, which can distort local embedding structures, therefore P3 is not maintained. Finally, retraining is required for updates, so P4 is not addressed.

Graph Neural Networks (GNNs) such as Graph Convolutional Networks (GCNs) (Kipf & Welling, 2017) and Graph Attention Networks (GATs) (Veličković et al., 2018) aggregate base embeddings ϕ^k with neighborhood contexts to produce richer representations ϕ^{k+1}, serving as final embeddings

$$\phi^{k+1}(v) = \sigma\big(W^k \phi^k(v) + \sum\nolimits_{u \in c(\mathcal{V}_\mathcal{A}, v)} \omega_{vu}^k \cdot W^k \phi^k(u)\big), \tag{1}$$

where $\phi^k(v)$ is the representation of node v at layer k, W^k is a learnable weight matrix, ω_{vu}^k are aggregation coefficients (uniform for GCNs and attention-based for GATs), σ is a nonlinearity, and $\mathcal{A} = \mathcal{V}$ holds. Extensions for KGs include RGCN (Schlichtkrull et al., 2017) and RGAT (Busbridge et al., 2019). GNNs require self-supervised signals like edge reconstruction or contrastive objectives for training. They do not satisfy P1 since random initialization of base embeddings ϕ^0 can lead

to divergent representations ϕ^k for structurally identical nodes. P2 and P3 depend on the chosen training signal, while P4 is only partially addressed, as aggregation can be recomputed but training is not designed for dynamic updates. Notably, Navi adapts GNNs by ignoring the self-term $W^k\phi^k(v)$ and by reconstructing embeddings solely from adjacent edges (Krause, 2022), which ensures P1 and P4 hold. However, Navi relies on fixed, pre-trained GEs ϕ^0, violating P2 and P3, and making performance dependent on their quality. In the following, we introduce LoRE, an end-to-end GE framework that extends this idea of local reconstruction while addressing all properties P1–P4.

4 Methodology

Building on the ideas of Navi, we introduce LoRE to extend translational GE paradigms with attentive, GNN-style local reconstructions. We first define the general model architecture, followed by its training objective, including a perspective-preserving cosine loss that respects local similarities.

4.1 Local Node Embedding Reconstructions via LoRE

Given an attributed node set $\mathcal{V}_\mathcal{A}$, LoRE considers two node embeddings $\phi, \psi : \mathcal{V} \to \mathbb{R}^d$. The base embedding ϕ is randomly initialized and ℓ^2-normalized throughout training, i.e., $\|\phi(v)\|_2 = 1$. In contrast, the independent node embedding $\psi(v) = \pi(c_{\mathcal{V}_\mathcal{A}}(v))$ represents a local reconstruction of $\phi(v)$, i.e., $\psi(v) \approx \phi(v)$, solely based on its attributes $c_{\mathcal{V}_\mathcal{A}}(v)$. Hence, ψ is an independent node embedding and attribute-invariant as such, fulfilling key property P1. However, this reconstruction requires a projection $\pi : 2^\mathcal{A} \to \mathbb{R}^d$ as per Definition 6, which LoRE instantiates via self-attention pooling of intermediate attribute embeddings $\Phi : \mathcal{A} \to \mathbb{R}^d$ with scaled dot-product weights:

$$\pi(A) = \tanh\left(\frac{1}{N}\sum\nolimits_{i=1}^{N}\left(\sum\nolimits_{j=1}^{N}\alpha_{ij}\,\Phi(a_j)\right)\right)$$

Here, $A = \{a_1, \ldots, a_N\}$ is a set of attributes and each entry a_i is transformed into an intermediate representation $\Phi(a_i) \in \mathbb{R}^d$. These are then mapped to queries $q_i = W_q\,\Phi(a_i)$ as well as keys $k_i = W_k\,\Phi(a_i)$, where $W_q, W_k \in \mathbb{R}^{d\times d}$ are learnable matrices. Scaled dot-products compute pairwise similarity scores between attributes and a row-wise softmax yields attention weights

$$\alpha_{ij} = \left(\sum\nolimits_{l=1}^{N}\exp\left(\frac{q_i^\top k_l}{\sqrt{d}}\right)\right)^{-1}\cdot\exp\left(\frac{q_i^\top k_j}{\sqrt{d}}\right).$$

The inner sum $\sum_{j=1}^{N}\alpha_{ij}\Phi(a_j)$ is the self-attention update for each attribute a_i, obtained as a weighted combination of all intermediate attribute embeddings. The outer average then pools these updated representations into a single vector summarizing the entire attribute set, and the hyperbolic tangent activation bounds this pooled vector within $(-1, 1)^d$, keeping $\pi(A)$ consistent with the ℓ^2-normalized base embedding $\phi(v)$ while preserving reconstruction flexibility and maintaining smooth gradients during training. However, while this provides an expressive attention mechanism to aggregate intermediate attribute embeddings, the latter must first be defined per use case. In the following, we propose a translational method to derive such embeddings for common graph structures.

4.2 Local Graph Embedding Reconstructions via LoRE

In the remainder of this work, we assume that $\mathcal{V}_\mathcal{A}$ induces a graph $\mathcal{G} = (\mathcal{V}, \mathcal{V}_\mathcal{A})$, i.e., $\mathcal{A} = \mathcal{V}\times\mathcal{M}$. Depending on the graph type, different implementations of the intermediate attribute embedding can be considered. We restrict ourselves to graphs that can be directed and/or labeled with relation types. If both characteristics are fulfilled, we are concerned with a KG and thus we adopt the idea of TransE due to its simplicity and the considerably fewer parameters learned per relation compared to the weight matrices of standard GNNs. For a directed, labeled graph with $\mathcal{M} = \mathcal{O}\times\mathcal{R}$, define

$$\Phi(u, o, r) = \phi(u) + \xi(o)\cdot\rho(r), \quad \xi(o) = \mathbb{1}_{\{o=o_{\text{in}}\}} - \mathbb{1}_{\{o=o_{\text{out}}\}} \in \{-1, 1\},$$

with trainable embeddings $\rho : \mathcal{R} \to \mathbb{R}^d$ which are not normalized to increase training flexibility.

For undirected LGs $\mathcal{G}$ with $\mathcal{A} = \mathcal{V} \times \mathcal{R}$, we use $\Phi(u, r) = \phi(u) + \rho(r)$. For unlabeled graphs, a single translational vector $\tau \in \mathbb{R}^d$ is used, giving $\Phi(u, o) = \phi(u) + \xi(o) \cdot \tau$ when $\mathcal{G}$ is directed and $\Phi(u) = \phi(u) + \tau$ when it is undirected. Finally, the embedding reconstruction $\psi(v)$ is obtained as

$$\psi(v) = \pi(c_{\mathcal{V}_\mathcal{A}}(v)) = \tanh\left(|c_{\mathcal{V}_\mathcal{A}}(v)|^{-1} \cdot \sum_{i=1}^{|c_{\mathcal{V}_\mathcal{A}}(v)|} \left(\sum_{j=1}^{|c_{\mathcal{V}_\mathcal{A}}(v)|} \alpha_{ij}\, \Phi(a_j)\right)\right).$$

While this formulation specifies how node representations are reconstructed from their attributes, it does not yet define how the trainable parameters should be optimized. Therefore, we introduce a perspective-preserving training objective designed to address the challenges of locality-awareness and OWA-consistent negative sampling to fulfill the key properties P2 and P3.

4.3 Training LoRE

Most GE methods rely on edge-level negative sampling. Given a node $v \in \mathcal{V}$ and one of its adjacent attributes $a = (u, m) \in \mathcal{A} = \mathcal{V} \times \mathcal{M}$, a negative node $\tilde{v}$ is sampled under the assumption that $a \notin c_{\mathcal{V}_\mathcal{A}}(\tilde{v})$ implies a false fact. Training then reduces to a single-edge reconstruction of $\phi(v)$ from (u, m), minimizing its distance to $\phi(v)$ while maximizing its distance to $\phi(\tilde{v})$. Under the OWA, this assumption is brittle, since the graph may simply be missing this edge for $\tilde{v}$ even if it holds in reality. In contrast, LoRE adopts the idea of Navi and reconstructs from the full neighborhood $c_{\mathcal{V}_\mathcal{A}}(v)$ instead of a single edge, thereby lowering the risk of OWA violations because two distinct nodes are far less likely to share all attributes than to share a single attribute. However, rather than assuming fixed base embeddings ϕ, LoRE learns them end-to-end. Accordingly, given a node $v \in \mathcal{V}$ and the embeddings $\phi(v), \psi(v)$, we sample another node $\tilde{v} \neq v$ and treat its base embedding $\phi(\tilde{v})$ as the negative reconstruction target. For instance, the cosine distance $\delta_{\cos}$ is a widely used distance measure to capture the relative orientation between two vectors:

$$\delta_{\cos}(x, x') = 1 - (x \cdot x') \cdot (\|x\|_2 \cdot \|x'\|_2)^{-1} \in [0, 2].$$

In contrast to ℓ^p distances, which are translation-invariant but sensitive to scaling, cosine distance is invariant under positive scalar multiplication but not translation-invariant. To incorporate both translation- and scale-invariance, thereby enabling locality-awareness, we define δ_{LoRE} as a perspective-preserving cosine distance for a prediction $\hat{y}$ and the positive/negative target $y, \tilde{y}$:

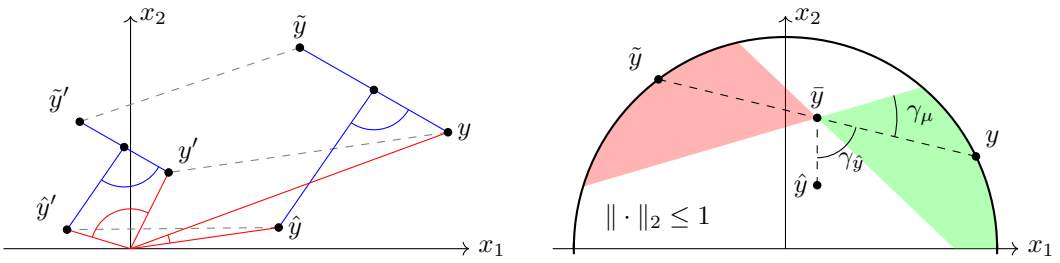
$$\delta_{\text{LoRE}}(\hat{y}, y; \tilde{y}) = \delta_{\cos}(\hat{y} - \bar{y}, y - \bar{y}) \quad \text{with} \quad \bar{y} := (y + \tilde{y})/2.$$

The point $\bar{y}$ serves as a local anchor that re-centers the prediction and the positive target with respect to the negative target before measuring their angular difference. This adjustment allows the distance to be both translation- and scale-invariant relative to the anchor, as illustrated in Figure 1a.

Standard MRL optimizations compare two distances and enforce a margin $\mu \geq 0$ between them. Even though LoRE considers a single distance relative to the negative target, a margin $\mu \geq 0$ can still be specified. This results in the setup depicted in Figure 1b, i.e., the margin μ corresponds to an angle γ_μ and predictions $\hat{y}$ with $\gamma_{\hat{y}} > \gamma_\mu$ are penalized. Given an angle $\gamma_\mu \in [0, \frac{\pi}{2})$, the corresponding margin μ can be determined via $\mu = 1 - \cos(\gamma_\mu)$. Due to the ℓ^2 normalization, the base embeddings ϕ are positioned on the unit sphere so the anchor $\bar{y}$ lies inside the unit ball. In conclusion, to account for locality-awareness as key property P3, LoRE is trained by minimizing

$$\lambda_{\text{LoRE}}(v, \tilde{v}) = \max\left(\delta_{\text{LoRE}}\left(\psi(v), \phi(v); \phi(\tilde{v})\right) - \mu, 0\right).$$

Two strategies are applied to improve training efficiency and generalization. First, batching is applied, i.e., M nodes together with their negative samples are processed. Second, a maximum attribute size $N_{\max}$ limits the number of edge attributes per node. Given $|c_{\mathcal{V}_\mathcal{A}}(v)| > N_{\max}$, we construct a reduced characteristic set $c'_{\mathcal{V}_\mathcal{A}}(v) \subseteq c_{\mathcal{V}_\mathcal{A}}(v)$ with $|c'_{\mathcal{V}_\mathcal{A}}(v)| \leq N_{\max}$ by randomly sampling at most $N_{\max}$ attributes. To prevent overfitting, this sampling is also applied for $|c_{\mathcal{V}_\mathcal{A}}(v)| \leq N_{\max}$, and the remaining $N_{\max} - |c'_{\mathcal{V}_\mathcal{A}}(v)|$ positions are masked within LoRE's attention mechanism.



(a) Visual representation of the translation- and scale-invariance of δ_{LoRE} (resulting in its locality-awareness) as a perspective-preserving distance (blue), compared to regular cosine distance (red).

(b) Geometric interpretation of the δ_{LoRE}-induced loss, relative to the target anchor $\bar{y}$. A prediction $\hat{y}$ is penalized if its angle $\gamma_{\hat{y}}$ to y exceeds γ_μ, thus preserving the perspective from the negative target $\tilde{y}$.

Figure 1: Visual representation and interpretation of δ_{LoRE} characteristics and training objectives.

5 Experimental Evaluation

LoRE's embedding quality is evaluated on established benchmark tasks. We focus on node classification tasks, i.e., predicting externally defined labels from pretrained embeddings. We therefore restrict our evaluation to external data mining tasks rather than internal link prediction. While both aim to predict missing information (Portisch et al., 2022), external labels provide a stable, verifiable ground truth, whereas link prediction depends on edge-level negative samples that can violate the OWA. Simple undirected graphs and KGs are considered as representative graph types. In each setting, we compare LoRE against established baselines, demonstrating that it satisfies the properties P1–P4 and generally outperforms existing methods, except when LoRE's neighborhood-invariance prevents deceptive overperformance. Analogous to standard GE benchmarks, embeddings are first trained self-supervised (labels are withheld), after which support vector machines (SVMs) and random forests (RFs) predict node labels. Embedding creation thus serves as a pretraining step. Additional analysis indicates that LoRE's self-reconstruction mechanism improves embedding generalization, explaining the observed gains. Finally, we evaluate its dynamic capabilities (P4), showing it surpasses Navi and GNNs on incremental update tasks simulating initially incomplete graphs.

LoRE embeddings of dimension $k = 256$ are trained using Adam with a fixed learning rate of $5 \cdot 10^{-5}$, a batch size of $M = 32$, and a maximum attribute count $N_{\max} = 16$. The δ_{LoRE} angle margin γ_μ is selected from $\{0, \frac{\pi}{6}\}$ via grid search (i.e., 0° or 30°), and a dropout rate of 0.2 is applied. Downstream SVM hyperparameters are also selected via grid search over regularization values $C \in \{0.01, 0.1, 1, 10, 100\}$. For RFs, the number of trees $\{100, 300, 500\}$ and maximum depths $\{10, 30, 50\}$ are considered. GEs and classifiers are trained independently ten times, and results are averaged. Experiments were run on an NVIDIA RTX 2080 Ti GPU with 12 GB VRAM. To ensure reproducibility, the PyTorch implementation of LoRE is publicly available[1].

5.1 Embedding Undirected Simple Graphs

We evaluate on the standard Planetoid citation networks Cora, CiteSeer, and PubMed (Yang et al., 2016) and the Amazon-Computers co-purchase graph from the SNAP benchmark (Yang & Leskovec, 2012), treating them as undirected SGs. These datasets are widely used for benchmarking GEs via downstream node classification, where nodes are papers or products and edges are citations or co-purchases. Dataset characteristics, including class imbalance, are summarized in Table 4 (Appendix A.1). We adopt the fixed Planetoid and SNAP splits for comparability with prior work. Baselines include HOPE (Ou et al., 2016), Node2Vec, GCN, and GAT, with the training configurations explained in Appendix A.2. Mean accuracies and standard deviations appear in Table 1, and macro-F1 scores in Table 6 (Appendix A.3.1) confirm the accuracy trends. GAT and LoRE achieve the highest accuracies, with LoRE showing the lowest deviations and thus the most stable overall performance. The training of HOPE and Node2Vec was the fastest but they performed worst. For GNN methods, a single LoRE forward pass took slightly longer than GCN or GAT, yet LoRE

[1] Anonymized repository with flask implementation: `https://github.com/LoRE-ICLR/LoRE`

needs only one layer by design, making it overall faster to train despite its self-attention mechanism. Memory differences among GNN-style approaches are minor since LoRE omits projection matrices. A generalized efficiency analysis that reinforces these findings can be found in Section 5.3.

	HOPE	Node2Vec	GCN	GAT	LoRE
Cora	81.73 ±1.62	82.21 ±1.48	83.18 ±1.66	85.11 ±2.05	84.88 ±0.95
CiteSeer	70.88 ±1.90	72.15 ±1.74	71.79 ±1.81	73.40 ±2.12	73.12 ±1.04
PubMed	79.64 ±1.72	79.12 ±1.55	80.73 ±1.63	81.95 ±2.08	82.21 ±1.14
Amazon	85.91 ±1.81	86.74 ±1.44	87.62 ±1.71	88.18 ±2.12	88.90 ±1.06

Table 1: Mean accuracies including standard deviations for the SG embedding benchmark tasks.

5.2 Embedding Knowledge Graphs

We use the benchmark KGs AIFB, MUTAG, BGS, and AM, which are widely adopted for evaluating KG embedding quality (Ristoski et al., 2016; Schlichtkrull et al., 2017). Each dataset provides a KG with literals and a predefined train–test split of node URIs and external class labels. Within our experiments, boolean literals are converted into instance–class relationships introducing two new classes per boolean property (true and false), while non-boolean literals are removed to match the contextual KG definition in Section 2. Dataset characteristics are summarized in Table 5 (Appendix A.1). Baselines include RESCAL, TransE, TransR, TransD, RDF2Vec, RGCN, and RGAT, with the training configurations from Appendix A.2. Mean accuracies with standard deviations are shown in Table 2. Again, macro-F1 scores in Table 7 (Appendix A.3.1) confirm the accuracy trends.

	RESCAL	TransE	TransR	TransD	RDF2Vec	RGCN	RGAT	LoRE
AIFB	84.10 ±1.28	87.22 ±1.63	85.56 ±1.72	71.39 ±1.88	88.89 ±1.77	91.30 ±1.22	92.22 ±0.93	91.67 ±0.00
MUTAG	70.50 ±1.52	75.00 ±1.81	69.44 ±1.74	67.50 ±1.96	73.68 ±1.83	75.29 ±1.48	83.97 ±1.08	85.29 ±0.91
BGS	67.80 ±1.67	69.31 ±1.89	68.28 ±1.92	69.31 ±1.85	68.28 ±1.94	72.41 ±1.39	73.45 ±1.11	74.14 ±1.03
AM	71.25 ±1.63	74.75 ±1.79	73.79 ±1.84	79.95 ±1.91	87.47 ±1.68	91.26 ±1.26	92.42 ±0.88	92.93 ±0.96

Table 2: Mean accuracies including standard deviations for the KG embedding benchmark tasks.

Once more, RGAT and LoRE achieve the highest accuracies overall, with LoRE slightly outperforming RGAT on MUTAG, BGS, and AM. However, the apparent RGAT lead on AIFB is not meaningful since ten of the 36 test nodes are topologically indistinguishable, and by design LoRE's neighborhood-invariance produces identical embeddings for such nodes, achieving the theoretical maximum accuracy of 91.67% by predicting the majority class. RGAT's higher score results from random variation and does not reflect a genuine advantage, as further training brings RGAT to the same value. Beyond these quantitative results, qualitative inspection of the embeddings confirms LoRE's improved separability over the baselines throughout the experiments. Appendix A.4, Figure 3, exemplifies this for the AIFB task, where LoRE produces distinctly separated and compact clusters compared to TransE and RGAT. Moreover, unlike RGCN and RGAT, which store a full projection matrix for each relation, LoRE represents each relation as a vector. Consequently, as the number of relations increases, LoRE becomes increasingly efficient in terms of training parameters and memory. This effect will be examined in detail in the generalized analysis of Section 5.3.

5.3 Embedding Efficiency and Dynamics

We analyze and simulate the memory efficiency and scalability of LoRE's self-reconstruction layers by comparing them to GCN and GAT as representative GNN layers. We consider a randomly generated graph $\mathcal{G}$ with nodes $\mathcal{V}$ and relation types $\mathcal{R}$, setting $|\mathcal{R}| = 1$ for unlabeled graphs. All methods require $|\mathcal{V}| \cdot d$ base embedding parameters. For GNNs, each additional layer introduces at least $|\mathcal{R}| \cdot d^2$ parameters for relation-specific projection matrices. In contrast, LoRE represents relations as vectors, and together with its query and key matrices for self-attention, we receive

$$param(\text{GNN}) \geq |\mathcal{V}| \cdot d + |\mathcal{R}| \cdot d^2 \quad \text{and} \quad param(\text{LoRE}) = (|\mathcal{V}| + |\mathcal{R}|) \cdot d + 2 \cdot d^2.$$

Solving $param(\text{GNN}) \geq param(\text{LoRE})$ yields $|\mathcal{R}| \geq 2 \cdot \frac{d}{d-1}$, meaning that for typical dimensions such as $d = 256$, two relation types already result in similar parameter counts. Every additional relation type increases parameters of existing GNNs compared to those of LoRE. Accordingly, forward

and backpropagation times for random directed graphs scale steeply for GCN and GAT but remain nearly flat for LoRE (see also Figure 2 of Appendix A.3.2). Because LoRE achieves competitive performance with a single layer, its efficiency advantage widens further in multi-relational settings.

To evaluate dynamic adaptability as key property P4, we once again consider the KG benchmarks, comparing LoRE with RGCN, RGAT, and Navi as a dynamic KG embedding approach. Incremental updates are simulated by randomly removing roughly half of each test node's edges before training, while ensuring that at least one edge remains per node. After GE generation and classifier training, these edges are inserted back, and the models are evaluated again by computing new embeddings and applying the existing classifier. Each experiment is repeated ten times using the same edge deletions for all models to ensure comparability. As shown in Table 3, the initial performances of RGAT and LoRE are higher than those of RGCN and Navi. After reinsertion, RGCN and RGAT typically degrade, indicating sensitivity to structural changes. In contrast, Navi and LoRE both improve after reinsertion, with LoRE consistently outperforming Navi across datasets. While these results need to be considered preliminary, they already suggest that LoRE's reconstruction mechanism not only improves generalization but also enables stable, real-time embedding adaptation without costly retraining. However, further experiments would need to be conducted to also evaluate scenarios involving completely new nodes, node deletions, and updated relation types.

	RGCN Start	RGCN End	RGAT Start	RGAT End	Navi Start	Navi End	LoRE Start	LoRE End
AIFB	59.10 ±4.87	55.44 ±3.63	74.27 ±4.08	73.86 ±4.72	72.57 ±3.54	75.17 ±1.17	76.03 ±3.96	85.20 ±1.33
MUTAG	61.59 ±5.26	57.94 ±4.98	65.17 ±4.67	66.02 ±4.92	60.44 ±3.22	65.04 ±0.89	63.60 ±3.48	66.18 ±1.14
BGS	52.30 ±5.81	49.75 ±5.39	69.40 ±5.03	63.14 ±5.58	69.03 ±3.85	71.58 ±1.21	68.20 ±2.26	72.36 ±0.97
AM	61.01 ±4.33	60.80 ±4.44	62.18 ±4.41	59.96 ±4.18	55.23 ±3.37	58.50 ±1.24	60.79 ±3.59	65.19 ±1.19

Table 3: Mean accuracies including standard deviations before (Start) and after (End) real-time embedding adaptations with updated neighborhood information.

6 Conclusion

In this work, we presented LoRE, a novel graph embedding framework designed to fulfill the four key properties P1–P4 as essential enablers of real-world graph applications. Through locally self-reconstructed embeddings, LoRE enforces neighborhood-invariance (P1), enables learning under incomplete graphs (P2), preserves locality-awareness (P3), and integrates graph updates on the fly (P4). Together, these properties yield adaptive and robust embeddings that respect structural equivalences, local geometry, and the OWA. Thanks to its generalized formulation, LoRE can be adapted to a variety of graph types, including undirected graphs, labeled graphs, and knowledge graphs, making it a versatile choice for diverse domains. Our experiments confirm that LoRE achieves competitive or superior accuracy compared to established baselines while maintaining parameter and memory efficiency. Its single-layer design, combined with relation-level vector representations, enables scalability and stable performance even as the number of relations grows.

While P1, P2, and P3 are satisfied by design, P4 concerning dynamic adaptability remains a promising direction for future research. Preliminary results indicate that LoRE performs strongly in simulated dynamic knowledge graph scenarios, exhibiting improvements after graph updates without requiring costly retraining. These findings highlight the potential of reconstruction-based embeddings for incremental learning in evolving graphs. A comprehensive evaluation of LoRE on large-scale graphs such as DBpedia (Auer et al., 2007) or Wikidata (Vrandečić & Krötzsch, 2014), including scenarios with new nodes, node deletions, relation-type updates, and robustness under noisy or adversarial updates, could further substantiate its practical applicability.

Future work may also explore integrating LoRE with downstream tasks beyond node classification, including link prediction, graph completion, and graph-level reasoning, as well as combining it with other self-supervised objectives to improve generalization. Moreover, hybrid architectures that embed LoRE's reconstruction mechanism within deeper GNN stacks or multi-view learning frameworks may yield additional benefits beyond those demonstrated so far.

Overall, the combination of theoretical grounding, flexibility, empirical performance, robustness, and dynamic adaptability positions LoRE as a promising general-purpose embedding method for dynamic, heterogeneous, and open-world graph environments.

REFERENCES

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. DBpedia: a nucleus for a web of open data. In *Proceedings of the 6th International Semantic Web Conference*, ISWC'07, pp. 722–735, 2007.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, NIPS'13, pp. 2787–2795, 2013.

Dan Busbridge, Dane Sherburn, Pietro Cavallo, and Nils Y. Hammerla. Relational graph attention networks, 2019. URL https://arxiv.org/abs/1904.05811.

Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering*, 31:833–852, 2017.

Genet Asefa Gesese, Russa Biswas, Mehwish Alam, and Harald Sack. A survey on knowledge graph embeddings with literals: Which model links better literal-ly? *Semantic Web*, 12(4):617–647, 2021.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 855–864. ACM, 2016.

Sang Hyu Hahn and Heiko Paulheim. Rdf2vec embeddings for updateable knowledge graphs – reuse, don't retrain! In *The Semantic Web: ESWC*, pp. 217–222, 2024.

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia D'amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge graphs. *ACM Comput. Surv.*, 54(4), 2021.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 687–696, 2015.

Hidetaka Kamigaito and Katsuhiko Hayashi. Comprehensive analysis of negative sampling in knowledge graph representation learning. In *Proceedings of the 39th International Conference on Machine Learning*, Proceedings of Machine Learning Research, pp. 10661–10675. PMLR, 2022.

Shima Khoshraftar and Aijun An. A survey on graph representation learning methods. *ACM Trans. Intell. Syst. Technol.*, 15(1), 2024.

Thomas N. Kipf and Max Welling. Variational graph auto-encoders, 2016. URL https://arxiv.org/abs/1611.07308.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations*, 2017.

Franz Krause. Dynamic knowledge graph embeddings via local embedding reconstructions. In *The Semantic Web: ESWC*, pp. 215–223, 2022.

Franz Krause, Tobias Weller, and Heiko Paulheim. On a generalized framework for time-aware knowledge graphs. In *Proceedings of the 18th International Conference on Semantic Systems /SEMANTiCS)*, volume 55, pp. 69–74, 2022.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pp. 2181–2187, 2015.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on Machine Learning)*, pp. 809–816, 2011.

Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2016.

Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 1105–1114. ACM, 2016.

Jan Portisch and Heiko Paulheim. The rdf2vec family of knowledge graph embedding methods: An experimental evaluation of rdf2vec variants and their capabilities. *Semantic Web*, 15(3):845–876, 2024.

Jan Portisch, Nicolas Heist, and Heiko Paulheim. Knowledge graph embedding for data mining vs. knowledge graph embedding for link prediction – two sides of the same coin? *Semantic Web*, 13 (3):399–422, 2022.

Petar Ristoski and Heiko Paulheim. Rdf2vec: Rdf graph embeddings for data mining. In *The Semantic Web – ISWC 2016*, pp. 498–514, 2016.

Petar Ristoski, Gerben Klaas Dirk de Vries, and Heiko Paulheim. A collection of benchmark datasets for systematic evaluations of machine learning on the semantic web. In *The Semantic Web – ISWC 2016*, pp. 186–194, 2016.

Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks, 2017. URL `https://arxiv.org/abs/1703.06103`.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations*, 2019.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48, pp. 2071–2080, 2016.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledge base. *Commun. ACM*, 57(10):78–85, 2014.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12): 2724–2743, 2017.

Mengjia Xu. Understanding graph embedding methods and their applications. *SIAM Review*, 63(4): 825–853, 2021.

Jaewon Yang and Jure Leskovec. Defining and evaluating network communities based on ground-truth. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, 2012.

Leshanshui Yang, Clément Chatelain, and Sébastien Adam. Dynamic graph representation learning with neural networks: A survey. *IEEE Access*, 12:43460–43484, 2024.

Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, ICML'16, pp. 40–48, 2016.

Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. Learning fair representations. In *Proceedings of the 30th International Conference on Machine Learning*, pp. 325–333, 2013.

Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.

A Appendix

A.1 Evaluation Datasets

The SG benchmarks in Table 4 include citation networks (Cora, CiteSeer, PubMed), where nodes represent papers and edges represent citation links, as well as the Amazon-Computers co-purchase network, where nodes correspond to products and edges indicate frequent co-purchases. These graphs vary in size and label diversity, providing a broad evaluation spectrum for node classification.

	Nodes	Edges	Classes	Train	Validation	Test
Cora	2,708	5,278	7	140	500	1,000
CiteSeer	3,327	4,552	6	120	500	1,000
PubMed	19,717	44,324	3	60	500	1,000
Amazon	13,752	245,861	10	8,251	2,750	2,751

Table 4: Dataset statistics for undirected graph benchmarks. Nodes and edges refer to the full graphs; Train/Validation/Test indicate the splits used for node classification. Class balances are color-coded.

The KG benchmarks in Table 5 cover diverse domains. AIFB and MUTAG are smaller academic and chemical datasets, while BGS (geological survey data) and AM (museum catalogue) are larger and more complex. The parenthesized values correspond to the original graphs before non-boolean literals were removed, whereas the first values indicate the filtered graphs used in our experiments.

	Nodes	Relations	Edges	Classes	Train	Test
AIFB	2,835 (8,285)	22	20,264 (29,043)	4	140	36
MUTAG	22,506 (23,644)	18	64,494 (74,227)	2	272	68
BGS	103,055 (333,845)	62	529,945 (916,199)	2	117	29
AM	958,232 (1,666,764)	45	3,299,143 (5,988,321)	11	802	198

Table 5: Dataset statistics before and after removing non-boolean literals (original values are added in parentheses). Boolean literals are retained as new nodes. Class balances are color-coded.

A.2 Baseline Configurations

As baseline models, we consider HOPE, RESCAL, Node2Vec, RDF2Vec, TransE, TransR, TransD, (R)GCN, (R)GAT, and Navi. All embeddings were ℓ^2-normalized and fixed to a dimensionality of $d = 256$. For GCN- and GAT-based baselines, GEs were first pretrained in a self-supervised manner using standard edge reconstruction with sigmoid cross-entropy loss (Kipf & Welling, 2016). Hyperparameters were selected using the performance of a downstream RF with 100 trees and depth 30. We tuned the number of message-passing layers $2, 3$ and, for GAT/RGAT, the number of attention heads $1, 2$. For KGs, this approach is relation-aware, predicting the validity of directed triples, while for SGs, all edges are treated as undirected and unlabeled. HOPE and RESCAL were tuned over ranks $64, 128, 256$ and regularization strengths $10^{-5}, 10^{-4}, 10^{-3}$ to match embedding dimensionality and prevent overfitting. Training batches for translational models consist of randomly sampled edge subsets of size 256 for SGs, AIFB, and MUTAG, and 512 for BGS and AM, without enforcing connectivity. Hyperparameters for translational models were selected via grid search over MRL margin values $0.5, 0.75, 1.0$. Node2Vec and RDF2Vec were tuned over walk lengths $4, 6, 8$, walks per node $10, 20$, and window sizes $5, 10$. The KG benchmarks do not provide validation splits and thus hyperparameters were selected via nested cross-validation. All baseline models were trained using the Adam optimizer with learning rates selected from $10^{-2}, 5 \cdot 10^{-3}, 10^{-3}, 5 \cdot 10^{-4}, 10^{-4}$, for 100 epochs with early stopping on validation performance. Finally, Navi used TransE embeddings as fixed base GEs for its initial node representations.

A.3 Additional Evaluation Material

A.3.1 Macro-F1 Scores

Tables 6 and 7 report macro-F1 scores for the SG and KG embedding benchmarks, providing a class-balanced view of performance. For all SG datasets, the macro-F1 values mirror the accuracy trends in Table 1: GAT and LoRE achieve the best overall performance, with LoRE showing consistently lower variance. Similarly, the KG results align with the accuracy outcomes in Table 2: RGAT and LoRE lead across most datasets, with LoRE slightly outperforming RGAT on MUTAG, BGS, and matching it on AIFB. These macro-F1 results reinforce that LoRE's performance advantage is consistent across both accuracy and class-balanced evaluation metrics.

	HOPE	Node2Vec	GCN	GAT	LoRE
Cora	80.12 ±1.70	80.76 ±1.56	81.89 ±1.63	83.84 ±2.03	83.66 ±0.99
CiteSeer	68.41 ±1.96	69.83 ±1.78	69.21 ±1.83	71.40 ±2.18	71.15 ±1.11
PubMed	77.85 ±1.79	77.52 ±1.63	78.60 ±1.70	79.62 ±2.10	79.18 ±1.16
Amazon	84.03 ±1.84	84.77 ±1.51	85.66 ±1.74	86.30 ±2.15	86.82 ±2.08

Table 6: Macro-F1 scores including standard deviations for the SG embedding benchmark tasks.

	RESCAL	TransE	TransR	TransD	RDF2Vec	RGCN	RGAT	LoRE
AIFB	80.45 ±1.33	81.67 ±1.70	79.92 ±1.77	69.84 ±1.93	84.15 ±1.82	87.12 ±1.28	90.20 ±0.96	88.95 ±0.08
MUTAG	68.72 ±1.61	73.49 ±1.86	67.82 ±1.79	65.93 ±2.02	72.04 ±1.89	74.11 ±1.54	82.10 ±1.14	83.45 ±0.98
BGS	66.05 ±1.74	67.63 ±1.95	66.54 ±1.97	67.95 ±1.89	66.80 ±1.99	71.05 ±1.44	72.18 ±1.17	73.01 ±1.09
AM	70.54 ±1.70	68.81 ±1.85	70.42 ±1.89	76.96 ±1.96	75.60 ±1.73	82.83 ±1.31	78.01 ±0.92	81.53 ±1.01

Table 7: Macro-F1 scores including standard deviations for the KG embedding benchmark tasks.

A.3.2 Comparison of Forward and Backpropagation Times

To obtain these measurements reported in Figure 2, we simulated forward passes and corresponding backpropagation steps with Adam for $|\mathcal{R}| = 5, 10, \ldots, 50$ relation types. For each configuration, $|\mathcal{R}|$ random node embeddings (embedding dimension 256) were generated, assuming one incoming edge per relation type. Each experiment was repeated ten times for each number of relations and graph embedding model, and averages were recorded. The results demonstrate that the two-layer RGCN and RGAT variants exhibit quasi-linear scaling in computation time, whereas LoRE maintains nearly flat scaling and therefore offers superior efficiency in multi-relational scenarios.

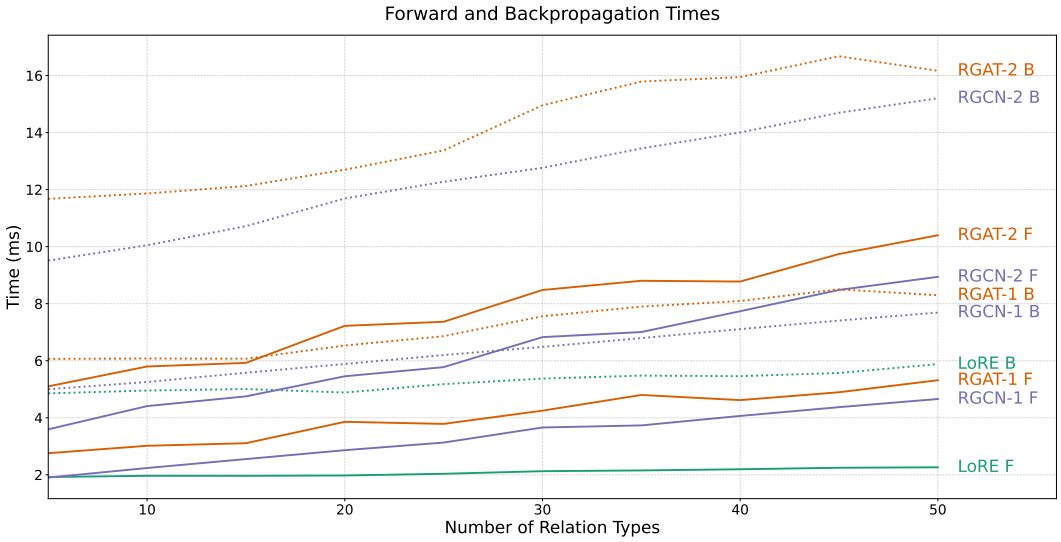
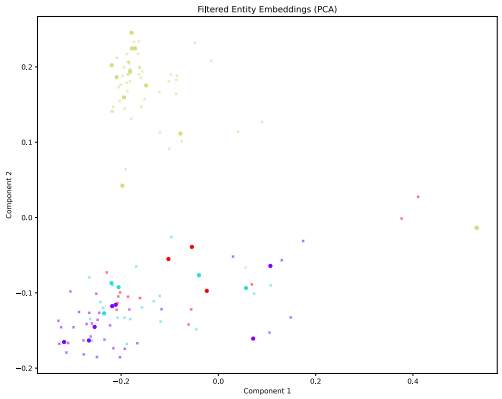


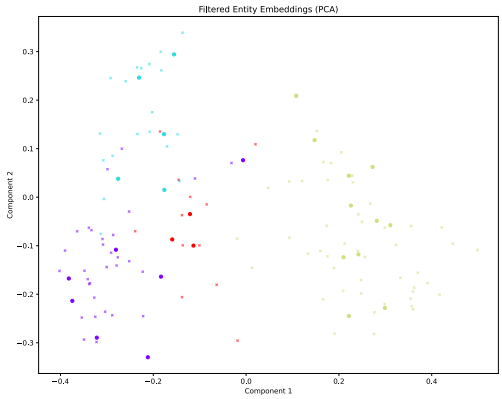
Figure 2: Forward and backpropagation times for LoRE, RGCN, and RGAT models across varying numbers of relation types. Solid lines represent forward passes, while dotted lines represent backpropagation using Adam. For RGCN and RGAT, both one-layer and two-layer variants are shown.

A.4 EMBEDDING CHARACTERISTICS

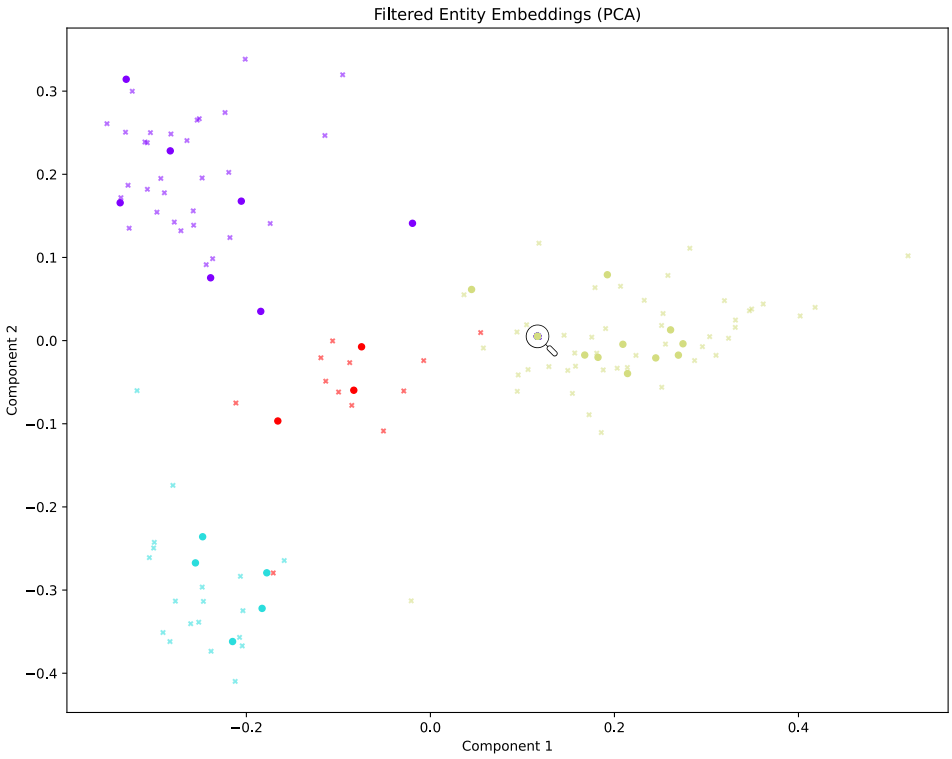
The visualizations in Figure 3 show that LoRE embeddings achieve clearer class separation compared to both TransE and RGAT, with more compact clusters and less overlap. While this effect is most pronounced on the AIFB dataset, similar patterns were observed on other benchmarks, suggesting that LoRE consistently yields more structured and discriminative representations. The magnifying glass highlights structurally equivalent nodes from three different classes. Due to LoRE's neighborhood-invariance, they are mapped within the majority cluster but positioned near its boundary, which is a desirable outcome for balancing structural consistency with class distinctions.



(a) PCA reduction of TransE embeddings.



(b) PCA reduction of RGAT embeddings.



(c) PCA reduction of LoRE embeddings.

Figure 3: PCA embedding visualizations of the AIFB task, where four classes are predicted. The PCA plots project the embeddings into 2 dimensions, showing the separability and clustering characteristics for (a) TransE, (b) RGAT, and (c) LoRE. Note: Test node embeddings are printed in bold.