Evaluation of Whole-Graph Embedding Methods on Real-World Graph Classification Benchmarks

Angeliki Spanou-Kapantoni*, Charalambos-Ioannis Sfiris*
* School of Electrical and Computer Engineering
National Technical University of Athens
Information Systems Course
A.M.: 03121090, 03121044
Emails: angeliki.kapantoni@gmail.com, harris.sfiris@gmail.com

***Abstract*—Whole-graph embedding methods transform variable-size graphs into fixed-length vectors, enabling the use of standard machine learning models for graph-level tasks. Despite their widespread adoption, the practical trade-offs between unsupervised structural descriptors and supervised graph neural networks remain insufficiently characterized beyond predictive accuracy.**

In this work, we present a systematic empirical benchmark of three representative paradigms and we evaluate the methods across multiple complementary criteria. Our results reveal clear and consistent trade-offs between method/dataset combinations. Overall, no single method dominates across all criteria; instead, each approach excels under different accuracy, efficiency, and robustness requirements.

***Index Terms*—Graph Embeddings, Graph Classification, Graph2Vec, NetLSD, Graph Neural Networks, GIN, TUDataset, Benchmarking, Robustness**

I. Introduction

Graphs constitute a natural and expressive data structure for modeling relational information, capturing interactions between entities in domains such as chemistry, biology, and social networks. In many practical applications, the goal is to perform *graph-level prediction*, where entire graphs must be classified according to their structural or semantic properties (e.g., predicting molecular mutagenicity or categorizing social interaction networks). A fundamental challenge in this setting is to transform variable-size graphs into fixed-length representations that are compatible with standard machine learning algorithms.

Early approaches relied on handcrafted features and graph kernels, which compare graphs through substructure counting or similarity functions. While often effective, such methods can be computationally expensive, difficult to scale, and require careful engineering for each domain. To overcome these limitations, *whole-graph embedding* techniques have been proposed, mapping each graph to a low-dimensional vector that preserves relevant structural information and enables efficient downstream learning.

Modern embedding methods broadly fall into two categories. *Unsupervised* approaches compute task-agnostic structural descriptors without using labels, offering simplicity and efficiency. In contrast, *supervised* approaches based on Graph Neural Networks (GNNs) learn task-specific representations end-to-end, typically achieving higher predictive performance at the cost of increased computational complexity and potential sensitivity to noise.

Despite the abundance of proposed methods, their practical trade-offs in terms of accuracy, clustering structure, computational efficiency, and robustness are not always clear. Many studies focus solely on predictive performance, leaving other critical aspects, such as computational cost, embedding geometry, and robustness under structural or feature noise, largely underexplored. As a result, practitioners lack systematic guidance on which embedding strategy is most appropriate under different resource and robustness constraints.

To address this gap, we present a comprehensive empirical benchmark of three representative whole-graph embedding paradigms: Graph2Vec and NetLSD (unsupervised) and the Graph Isomorphism Network (GIN) (supervised). We evaluate them on three widely used TUDataset benchmarks (MUTAG, ENZYMES, and IMDB-MULTI) under a unified experimental protocol that considers not only classification accuracy but also clustering behavior, runtime cost, and stability under controlled structural and feature perturbations, quantified through embedding drift and prediction degradation.

Our main contributions are summarized as follows:

- We provide a unified experimental comparison of unsupervised and supervised whole-graph embedding methods across multiple real-world datasets.
- We evaluate embeddings on classification accuracy, clustering quality, computational efficiency, and robustness under controlled structural and feature perturbations.
- We introduce a systematic stability analysis that measures both embedding drift (cosine similarity) and downstream accuracy degradation.
- We release reproducible code and scripts to facilitate fair comparison and future benchmarking.

II. Related Work

Whole-graph representation learning has been extensively studied through graph kernels, handcrafted descriptors, and, more recently, neural architectures. Classical graph kernels, such as Weisfeiler–Lehman (WL) subtree kernels, compare graphs by counting shared substructures and often achieve

strong accuracy, but they typically suffer from high computational cost and limited scalability to large datasets. To address these limitations, researchers have proposed vector-based graph descriptors that summarize structural properties into fixed-length signatures suitable for standard machine learning pipelines.

Unsupervised whole-graph embedding methods aim to learn task-agnostic representations without using labels. Graph2Vec [1] adapts ideas from document embedding by treating rooted subgraphs as "words" and graphs as "documents", learning dense representations through a skip-gram objective. NetLSD [2] takes a complementary spectral perspective and computes permutation-invariant signatures based on Laplacian heat kernel traces across multiple time scales, capturing both local and global structural characteristics. Such approaches are computationally efficient, deterministic, and easy to integrate with downstream classifiers.

More recently, Graph Neural Networks (GNNs) have become the dominant paradigm for graph representation learning. By iteratively aggregating information from node neighborhoods, GNNs learn task-specific embeddings optimized end-to-end for the target objective. The Graph Isomorphism Network (GIN) [3] is a particularly expressive architecture that matches the discriminative power of the Weisfeiler–Lehman test through learnable aggregation functions and multilayer perceptrons. GNN implementations are widely supported by libraries such as PyTorch Geometric [4], enabling scalable training on modern hardware.

Benchmark evaluation of graph embedding methods is commonly performed on standardized datasets such as those provided by TUDataset [5]. However, most prior work evaluates methods primarily through predictive accuracy, with limited analysis of complementary properties such as embedding geometry, computational efficiency, or robustness to structural and feature perturbations.

III. Methods

A. Problem Setting

Let $\{(G_i, y_i)\}_{i=1}^{N}$ denote a dataset of labeled graphs, where each graph $G_i = (V_i, E_i)$ consists of a variable number of nodes and edges and $y_i \in \{1, \ldots, C\}$ is a graph-level class label. The objective of graph classification is to learn a function $f : \mathcal{G} \to \mathcal{Y}$ that predicts the label of an unseen graph.

Whole-graph embedding methods address this task by mapping each graph to a fixed-dimensional representation $\mathbf{z}_i \in R^d$. These embeddings enable the use of standard machine learning models (e.g., SVMs or neural classifiers) while preserving structural and semantic properties of the original graph. In this work, all compared approaches ultimately produce such vectors $\mathbf{z}_i$ and are evaluated under the same downstream pipeline to ensure a fair comparison.

B. Graph2Vec

Graph2Vec [1] is an unsupervised representation learning method inspired by the Doc2Vec framework from natural language processing. Each graph is treated analogously to a document and decomposed into a multiset of rooted subgraphs extracted using the Weisfeiler–Lehman relabeling procedure. These subgraphs act as tokens describing local structural contexts.

A skip-gram objective is then optimized to maximize the likelihood of observing subgraph tokens given a graph identifier. Formally, the method learns embeddings by maximizing the co-occurrence probability between graphs and their rooted substructures. The resulting representations are dense, task-agnostic vectors that summarize structural patterns without using label information.

Graph2Vec is attractive due to its simplicity, scalability, and independence from supervision, but it does not provide a stable out-of-sample inference mechanism without refitting the model.

C. NetLSD

NetLSD (Network Laplacian Spectral Descriptor) [2] computes permutation-invariant graph signatures based on the spectrum of the graph Laplacian. Let L denote the normalized Laplacian matrix. NetLSD defines a heat kernel trace

$$h(t) = \operatorname{tr}\left(e^{-tL}\right), \tag{1}$$

evaluated over multiple time scales t. This trace summarizes how heat diffuses through the graph, capturing structural information at both local (small t) and global (large t) scales.

By sampling $h(t)$ across a range of time values, a fixed-length signature vector is obtained. Because it depends only on eigenvalues, the descriptor is invariant to node permutations and deterministic, leading to highly stable embeddings. NetLSD is computationally efficient and naturally robust to small structural perturbations.

D. Graph Isomorphism Network (GIN)

The Graph Isomorphism Network (GIN) [3] is a supervised graph neural network designed to match the expressive power of the Weisfeiler–Lehman graph isomorphism test. GIN updates node representations through iterative neighborhood aggregation.

At layer k, each node v updates its embedding according to

$$\mathbf{h}_v^{(k)} = \mathrm{MLP}^{(k)}\left((1+\epsilon^{(k)})\mathbf{h}_v^{(k-1)} + \sum_{u \in \mathcal{N}(v)} \mathbf{h}_u^{(k-1)}\right), \tag{2}$$

where $\mathcal{N}(v)$ denotes neighbors of node v and $\epsilon^{(k)}$ controls the weighting between self- and neighbor-information.

After several layers, node embeddings are aggregated into a graph-level representation using a permutation-invariant readout function (global sum pooling in our implementation):

$$\mathbf{z}_G = \sum_{v \in V} \mathbf{h}_v^{(K)}. \tag{3}$$

TABLE I: Dataset Statistics.

Dataset	#Graphs	#Classes	Avg. #Nodes	Avg. #Edges
MUTAG	188	2	17.93	19.79
ENZYMES	600	6	32.63	62.14
IMDB-MULTI	1500	3	13.00	65.94

The graph embedding is then passed to a classifier and trained end-to-end using supervised loss. Compared to unsupervised descriptors, GIN can adapt representations directly to the target task, often achieving higher predictive performance at the cost of increased computational complexity and training time.

IV. Experimental Setup

A. Infrastructure and Software

Experiments were executed in Google Colab using an NVIDIA T4 GPU. The implementation was written in Python. Graph2Vec and NetLSD were obtained via the KarateClub library. GIN was implemented using PyTorch Geometric. We used Scikit-learn for SVM training, k-means clustering, and evaluation metrics.

B. Datasets

We used three graph classification datasets from TUDataset:

- **MUTAG**: small molecular graphs labeled by mutagenicity.
- **ENZYMES**: protein graphs categorized into enzyme classes.
- **IMDB-MULTI**: social graphs representing actor collaborations for movie genres.

These datasets cover diverse domains and vary substantially in graph size, number of classes, and structural complexity, providing a representative testbed for comparing embedding methods.

Table I summarizes dataset statistics.

C. Embedding Dimensionality and Comparison Setup

For Graph2Vec and NetLSD we used embedding dimensionality $d = 128$. For GIN, the graph representation was obtained by global sum pooling over node embeddings, resulting in a $d = 64$-dimensional graph embedding (hidden dimension). All methods were evaluated under the same train/test protocol and repeated across three random seeds (42, 43, 44). Although dimensionalities are not identical, all methods are evaluated under the same downstream protocols; thus the comparison focuses on practical performance rather than strictly matched representation size.

D. Classification Protocol

We performed graph classification experiments on MUTAG, ENZYMES, and IMDB-MULTI using three random seeds (42, 43, 44).

a) GIN: For each dataset, graphs were shuffled and split into 80% training and 20% test sets. If node attributes were missing, we applied the OneHotDegree transform, where node features are one-hot encodings of node degree. GIN was trained end-to-end using cross-entropy loss and the Adam optimizer. Hyperparameters (number of layers, dropout, learning rate, and weight decay) were tuned with Optuna (10 trials), using a short training budget during tuning, and then the model was retrained with the best configuration for 200 epochs. We report the best test performance observed during training.

b) Graph2Vec and NetLSD + SVM: For unsupervised methods, graphs were converted to NetworkX format and split into 80% training and 20% test sets. Embeddings were used as input features to an RBF-kernel SVM classifier. SVM hyperparameters (C, γ) were tuned with Optuna (10 trials) using an inner validation split of the training set. Final evaluation was performed on the held-out test set.

c) Remark on transductive unsupervised embeddings: For Graph2Vec and NetLSD, embeddings were learned in an unsupervised transductive manner: the embedding model was fitted on the full set of graphs (train + test) without using labels, while the SVM classifier was trained exclusively on the training split and evaluated on the test split.

d) Metrics: We report Accuracy, weighted F1-score, and ROC-AUC (one-vs-rest for multi-class when applicable). We also record embedding generation time, classifier training time, and Optuna tuning time.

E. Clustering Protocol and Visualization

We evaluated how well the learned graph embeddings form separable clusters without using labels for training. For each method (Graph2Vec, NetLSD, and GIN) and each dataset (MUTAG, ENZYMES, IMDB-MULTI), we applied two clustering algorithms on the precomputed graph-level embeddings: (i) k-means and (ii) Spectral Clustering with nearest-neighbor affinity. The number of clusters was set to the number of ground-truth classes for each dataset.

a) Repeated runs: All clustering experiments were repeated with three random seeds (42, 43, 44) to account for randomness in clustering initialization and visualization methods. We report mean and standard deviation across seeds.

b) Metrics: Clustering quality is primarily evaluated using Adjusted Rand Index (ARI), which measures the agreement between predicted clusters and ground-truth labels while correcting for chance. We additionally report Silhouette score as a complementary indicator of embedding-space compactness and separation; however, Silhouette does not directly measure alignment to class labels and may not correlate with ARI.

c) 2D Projections: To qualitatively inspect separability, we used t-SNE and UMAP to project embeddings into 2D and produced scatter plots colored by true class labels. For consistent comparisons, we use the same hyperparameters across methods and datasets (t-SNE perplexity 35, 1500 iterations; UMAP with 15 neighbors and min_dist=0.1).

F. Stability Analysis Under Perturbations

To assess robustness of whole-graph embeddings under controlled noise, we performed a stability analysis in which graphs were systematically perturbed and embeddings were recomputed. We then quantified (i) how much the resulting embeddings drift from the originals and (ii) how much downstream predictive performance degrades when the classifier is exposed to perturbed test graphs.

a) GIN (supervised, fixed checkpoint): GIN stability was evaluated using the trained checkpoints obtained from our classification experiments (saved per dataset and seed).

1) Perturbation Models: We considered structural perturbations affecting the edge set and, for GIN, an additional feature perturbation affecting node attributes.

a) Edge perturbations: For each graph $G = (V, E)$ we generated a perturbed graph $\tilde{G} = (V, \tilde{E})$ by:

- **Edge removal:** removing a fraction p_r of existing edges, chosen uniformly at random without replacement.
- **Edge addition:** adding a fraction p_a of new edges, sampled uniformly at random from the set of non-edges.

The number of removed edges was $k_r = \lfloor p_r|E| \rfloor$ and the number of added edges was $k_a = \lfloor p_a \max(1, |E|) \rfloor$, ensuring the perturbation strength scales with graph density. All graphs were treated as undirected simple graphs; for PyG graphs we explicitly rebuilt an undirected `edge_index` by adding both directions for each undirected pair.

Perturbations were generated deterministically using per-graph seeds of the form $s \cdot 10^4 + i$, where s is the experiment seed and i is the graph index in the ordered list. This guarantees that for a given seed and perturbation level, the same edges are perturbed every run.

b) Attribute shuffling: To probe sensitivity to node-feature noise, we also considered a feature perturbation in which node attributes within each graph were permuted by shuffling the rows of $X \in R^{|V| \times F}$:

$$\tilde{X} = PX,$$

where P is a random permutation matrix sampled deterministically per graph (seed $s \cdot 2 \cdot 10^4 + i$). This preserves the multiset of node features but destroys their assignment to specific nodes, providing a stress test for feature-dependent message passing. We ran GIN stability in two conditions: **no-shuffle** and **shuffle**.

2) Perturbation Sweeps: We evaluated robustness under two one-dimensional sweeps: (i) removal-only, where $p_a = 0$ and $p_r \in \{0.0, 0.05, 0.1, 0.2\}$, and (ii) addition-only, where $p_r = 0$ and $p_a \in \{0.0, 0.05, 0.1, 0.2\}$. Each configuration was repeated for seeds $\{42, 43, 44\}$.

*a) **Graph2Vec**:* Graph2Vec embeddings were learned using KarateClub with dimension $d = 128$, WL iterations 2, and 200 epochs. Because Graph2Vec requires refitting to embed a modified corpus and its training is stochastic, the resulting embeddings for the original and perturbed corpora may lie in different latent coordinate systems (up to rotations and other symmetries). Therefore, we treat Graph2Vec stability results as *pipeline sensitivity under refitting* rather than a strictly comparable embedding-drift measure in a fixed feature space.

*b) **NetLSD**:* NetLSD embeddings were computed using KarateClub (default configuration). As a deterministic spectral signature method, NetLSD has minimal stochasticity; nonetheless we kept the same pipeline structure and recomputed embeddings on the perturbed graphs for fairness.

*c) **GIN**:* For GIN, we extracted graph-level embeddings from the trained model by taking the pooled representation after the last message passing layer (global add pooling), *before* the final classifier MLP. Embeddings were computed for the full dataset in a single pass using `DataLoader` with `batch_size=32` and `shuffle=False`, both for original and perturbed graphs.

3) Stability Metrics: We quantified stability using both representation drift and downstream accuracy degradation.

a) Embedding stability (mean cosine similarity).: Let $Z^{(0)} \in R^{N \times d}$ denote the matrix of original graph embeddings and $Z^{(1)}$ the embeddings after perturbation (paired graph-by-graph). We define stability as the mean cosine similarity:

$$\mathrm{Stab}(Z^{(0)}, Z^{(1)}) = \frac{1}{N} \sum_{i=1}^{N} \frac{\langle z_i^{(0)}, z_i^{(1)} \rangle}{\|z_i^{(0)}\|_2 \, \|z_i^{(1)}\|_2}.$$

Higher values indicate that embeddings are invariant to perturbations. For Graph2Vec, this cosine similarity is computed between embeddings produced by two independently refit models and should be interpreted only as a heuristic indicator, since the two embedding sets are not guaranteed to be aligned in the same coordinate system.

b) Robustness of predictive performance (accuracy drop): We also measured the degradation in classification accuracy caused by perturbations:

$$\Delta\mathrm{Acc} = \mathrm{Acc}_{\mathrm{orig}} - \mathrm{Acc}_{\mathrm{pert}}.$$

For NetLSD, we trained a fixed SVM on the original training embeddings and evaluated it on original vs perturbed test embeddings, keeping the classifier fixed. For Graph2Vec, we do not interpret a fixed-classifier evaluation across perturbations as a valid test-time robustness measure; Graph2Vec results are reported separately as pipeline sensitivity (see Threats to Validity). For GIN, we evaluated the fixed checkpoint on the original test graphs and on the perturbed test graphs (with optional attribute shuffling), again without retraining.

V. Results

A. Graph Classification

Table II reports classification results as mean ± standard deviation over three random seeds (42, 43, 44). Across all datasets, GIN achieves the best overall predictive performance, especially on MUTAG and ENZYMES, indicating that supervised message passing learns task-relevant structural patterns more effectively than fixed unsupervised descriptors. On IMDB-MULTI, the performance gap is smaller, with Graph2Vec and GIN being closer in accuracy, suggesting that

TABLE II: Classification performance (mean ± std over 3 seeds).

Method	MUTAG	ENZYMES	IMDB-MULTI
	Accuracy (%)		
Graph2Vec	79.0 ± 0.0	38.3 ± 2.2	43.3 ± 1.7
NetLSD	80.7 ± 5.5	27.8 ± 4.6	46.4 ± 2.2
GIN	93.9 ± 3.0	47.5 ± 5.1	53.8 ± 1.8
	Weighted F1		
Graph2Vec	0.786 ± 0.007	0.379 ± 0.029	0.422 ± 0.008
NetLSD	0.801 ± 0.061	0.231 ± 0.061	0.430 ± 0.041
GIN	0.939 ± 0.030	0.475 ± 0.045	0.528 ± 0.020
	ROC-AUC		
Graph2Vec	0.870 ± 0.009	0.729 ± 0.005	0.615 ± 0.042
NetLSD	0.845 ± 0.015	0.606 ± 0.067	0.636 ± 0.020
GIN	0.958 ± 0.015	0.746 ± 0.013	0.708 ± 0.016

TABLE III: Clustering performance (ARI, mean ± std over 3 seeds). Higher is better.

Clustering	Method	MUTAG	ENZYMES	IMDB -MULTI
k-means	Graph2Vec	0.117 ± 0.007	0.018 ± 0.004	0.004 ± 0.004
	NetLSD	0.333 ± 0.034	0.015 ± 0.004	0.009 ± 0.006
	GIN	0.235 ± 0.095	0.029 ± 0.016	0.005 ± 0.007
Spectral	Graph2Vec	0.119 ± 0.032	0.015 ± 0.004	0.000 ± 0.000
	NetLSD	0.267 ± 0.000	0.026 ± 0.000	0.005 ± 0.003
	GIN	0.054 ± 0.175	0.042 ± 0.022	0.001 ± 0.001

simpler global patterns captured by unsupervised representations can remain competitive for social interaction graphs.

NetLSD performs competitively on MUTAG and provides stable results, while Graph2Vec shows slightly better performance than NetLSD on IMDB-MULTI. On ENZYMES, both unsupervised methods lag notably behind GIN, highlighting the increased difficulty of this multi-class dataset and the benefit of end-to-end learning.

B. Clustering Quality and Visualization

Table III reports clustering performance (ARI) for k-means and Spectral Clustering, averaged over three seeds. Overall, GIN embeddings yield the strongest label-aligned clustering on MUTAG (ARI $\approx$ 0.31 with k-means), indicating that supervised message passing produces representations that better separate classes in the embedding space. On ENZYMES and IMDB-MULTI, ARI values are substantially lower for all methods, suggesting limited class separability in an unsupervised clustering setting, even when classification performance is reasonable.

Interestingly, Silhouette scores (Table IV) can be high even when ARI is low, highlighting that embeddings may form compact clusters that do not correspond to the ground-truth classes. This reinforces ARI as the primary metric for label-aligned clustering evaluation.

Figure 1 provides representative t-SNE visualizations that qualitatively support the quantitative ARI findings. GIN embeddings on MUTAG form two clearly separable clusters that closely match the ground-truth labels. In contrast, Graph2Vec on ENZYMES produces largely overlapping embeddings with no discernible structure. Interestingly, NetLSD on IMDB-MULTI yields compact and well-separated geometric clusters despite poor label alignment, explaining the discrepancy between high Silhouette scores and near-zero ARI. This observation highlights that geometric separability in the embedding space does not necessarily imply class-discriminative representations.

TABLE IV: Silhouette scores (mean ± std over 3 seeds). Higher indicates better separation in embedding space, but not necessarily better class alignment.

Clustering	Method	MUTAG	ENZYMES	IMDB -MULTI
k-means	Graph2Vec	0.241 ± 0.083	0.039 ± 0.005	0.201 ± 0.029
	NetLSD	0.580 ± 0.010	0.434 ± 0.007	0.529 ± 0.015
	GIN	0.480 ± 0.155	0.156 ± 0.032	0.830 ± 0.183
Spectral	Graph2Vec	0.286 ± 0.010	0.030 ± 0.001	0.013 ± 0.009
	NetLSD	0.571 ± 0.000	0.280 ± 0.000	0.154 ± 0.084
	GIN	0.353 ± 0.080	0.135 ± 0.022	-0.385 ± 0.347

To further examine the global structure of the embedding spaces, we additionally visualize them using UMAP, which better preserves both local neighborhoods and large-scale manifold geometry. The UMAP projections corroborate the quantitative clustering results and reveal three characteristic behaviors across methods.

GIN on MUTAG forms two clearly separated clusters that closely align with the ground-truth labels, confirming that supervised message passing produces class-discriminative representations. NetLSD on ENZYMES, in contrast, generates compact and well-structured geometric clusters that remain largely label-mixed, explaining the discrepancy between high Silhouette scores and near-zero ARI. Finally, Graph2Vec on ENZYMES exhibits a diffuse and overlapping embedding cloud with no discernible structure, consistent with its weak clustering performance.

These observations reinforce that geometric compactness alone does not guarantee semantic separability and highlight the importance of supervision for aligning embedding geometry with class boundaries.

C. Stability Under Perturbations

We evaluate robustness of whole-graph embeddings under controlled perturbations of graph structure and (for GIN) node attributes. For each dataset and seed, we generate perturbed versions of all graphs and recompute embeddings, then quantify (i) embedding drift and (ii) degradation of downstream predictive accuracy. Embedding drift is measured as the mean cosine similarity between original and perturbed graph embeddings (higher is better). Predictive robustness is measured as accuracy drop $\Delta\mathrm{Acc} = \mathrm{Acc}_{\mathrm{orig}} - \mathrm{Acc}_{\mathrm{pert}}$ on the test split (lower is better). Results are averaged over three random seeds (42, 43, 44).

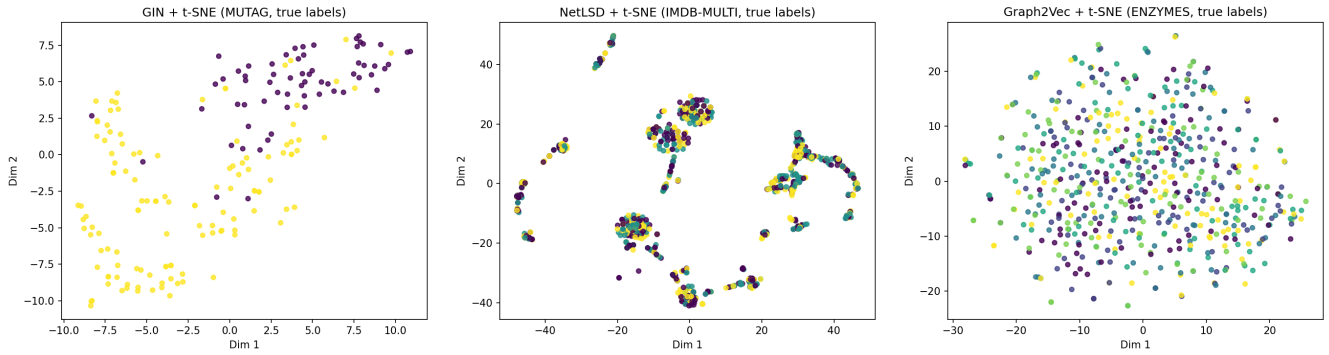

Fig. 1: Representative t-SNE projections of graph embeddings. Left: GIN on MUTAG shows clear class separation. Middle: NetLSD on IMDB-MULTI forms compact structural clusters but labels remain mixed. Right: Graph2Vec on ENZYMES exhibits poor separability.

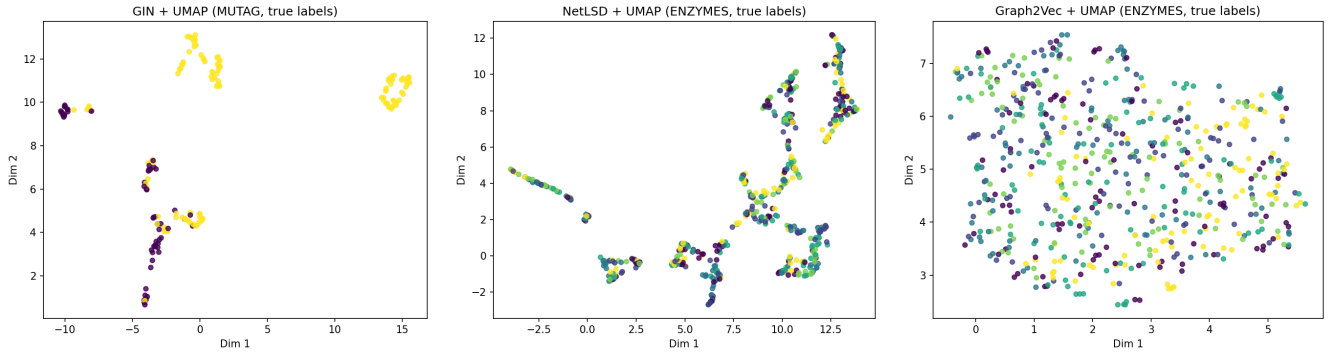

Fig. 2: Representative UMAP projections of graph embeddings. Left: GIN on MUTAG shows clear class-aligned separation. Middle: NetLSD on ENZYMES forms compact geometric clusters but labels remain mixed. Right: Graph2Vec on ENZYMES produces a diffuse embedding cloud with poor separability.

1) Overall trends: Figure 3 summarizes the accuracy drop as a function of perturbation strength for MUTAG, ENZYMES, and IMDB-MULTI. Several consistent patterns emerge.

(i) NetLSD is the most embedding-stable. Across all datasets and perturbation types, NetLSD exhibits near-perfect stability (cosine similarity $\approx$ 1), confirming that spectral signatures are highly invariant to moderate random edge edits. However, high stability does not always imply zero accuracy degradation: on MUTAG, NetLSD remains almost unchanged in cosine space, yet the SVM accuracy can still drop noticeably under edge removals, indicating that small (but systematic) shifts in specific spectral coordinates may still affect the decision boundary.

(ii) Graph2Vec shows moderate sensitivity. Graph2Vec shows moderate pipeline sensitivity under refitting: refitting on perturbed corpora changes the learned embedding space even for small perturbations; therefore we report Graph2Vec only as embedding drift (cosine) and do not report fixed-classifier accuracy drop.

(iii) GIN robustness depends strongly on features. For GIN with fixed checkpoints, edge perturbations alone produce dataset-dependent behavior: on MUTAG, removals can be particularly damaging, whereas on IMDB-MULTI the accuracy drop is close to zero (and occasionally negative, i.e., slight accuracy gains). When node attributes are additionally shuffled (GIN shuffle condition), robustness consistently deteriorates: both embedding similarity decreases substantially and accuracy drop increases, showing that message passing relies strongly on coherent feature-to-node assignments.

2) Quantitative summary: Table V reports stability (mean cosine similarity) together with $\Delta\mathrm{Acc}$ at perturbation level $p = 0.2$ for both removal-only and addition-only regimes, including both GIN variants (no-shuffle vs. attribute shuffle). For completeness, Tables VI–VIII report accuracy drop across all tested perturbation levels.

Graph2Vec reporting. Because Graph2Vec embeddings are recomputed by refitting on each perturbed corpus, the resulting coordinate systems are not guaranteed to be aligned across runs. Therefore, fixed-classifier accuracy drop is not a valid robustness metric. For this reason, we report only embedding cosine stability (pipeline drift) for Graph2Vec and omit $\Delta\mathrm{Acc}$ comparisons.

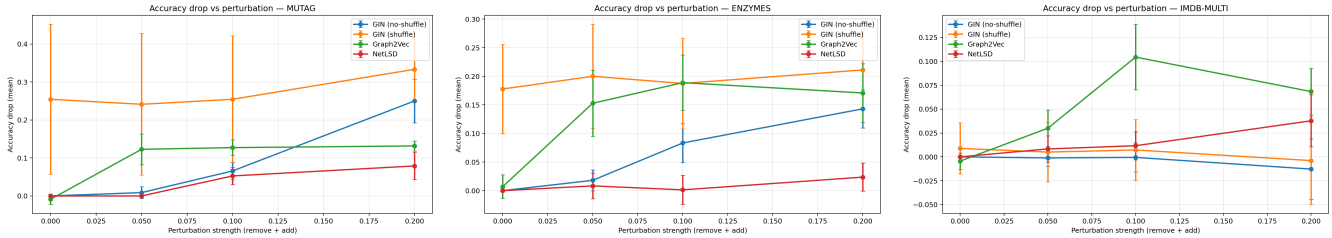

Fig. 3: Accuracy drop ΔAcc as a function of perturbation strength (p_r or p_a) for MUTAG, ENZYMES, and IMDB-MULTI. Error bars indicate standard deviation over three seeds. GIN is evaluated with fixed checkpoints under edge perturbations, with and without node-attribute shuffling.

TABLE V: Embedding stability (mean cosine similarity) and accuracy drop at perturbation level $p = 0.2$ (mean $\pm$ std over 3 seeds). Higher stability is better; lower ΔAcc is better. **Graph2Vec note:** embeddings are recomputed via refitting on each perturbed corpus, so ΔAcc is not a valid fixed-classifier robustness metric and is reported as N/A.

Dataset	Method	Removal-only ($p_r = 0.2$)		Addition-only ($p_a = 0.2$)	
		Stability ↑	ΔAcc ↓	Stability ↑	ΔAcc ↓
MUTAG	GIN (no-shuffle)	0.4393 ± 0.1901	0.3947 ± 0.0696	0.4594 ± 0.1927	0.1053 ± 0.0456
	GIN (shuffle)	0.2377 ± 0.2365	0.5175 ± 0.1096	0.4106 ± 0.0931	0.1491 ± 0.0608
	Graph2Vec	0.6103 ± 0.0056	N/A	0.6985 ± 0.0117	N/A
	NetLSD	0.9973 ± 0.0000	0.1404 ± 0.0402	0.9999 ± 0.0000	0.0175 ± 0.0304
ENZYMES	GIN (no-shuffle)	0.7029 ± 0.0796	0.1139 ± 0.0127	0.5404 ± 0.0700	0.1722 ± 0.0536
	GIN (shuffle)	0.4390 ± 0.1173	0.2000 ± 0.0726	0.3205 ± 0.0554	0.2222 ± 0.0668
	Graph2Vec	0.4420 ± 0.0015	N/A	0.4331 ± 0.0072	N/A
	NetLSD	0.9997 ± 0.0000	0.0167 ± 0.0363	0.9998 ± 0.0000	0.0306 ± 0.0127
IMDB-MULTI	GIN (no-shuffle)	0.9326 ± 0.0228	-0.0222 ± 0.0430	0.9497 ± 0.0178	-0.0033 ± 0.0203
	GIN (shuffle)	0.8132 ± 0.0478	-0.0178 ± 0.0570	0.8433 ± 0.0415	0.0100 ± 0.0348
	Graph2Vec	0.3698 ± 0.0285	N/A	0.4841 ± 0.0108	N/A
	NetLSD	1.0000 ± 0.0000	0.0578 ± 0.0329	1.0000 ± 0.0000	0.0178 ± 0.0217

TABLE VI: Accuracy drop ΔAcc under edge perturbations on MUTAG (mean $\pm$ std over 3 seeds). Lower is better.

	Removal-only (p_r, $p_a = 0$)				Addition-only (p_a, $p_r = 0$)			
Method	0.00	0.05	0.10	0.20	0.00	0.05	0.10	0.20
GIN (no-shuffle)	0.000 ± 0.000	0.018 ± 0.030	0.035 ± 0.015	0.395 ± 0.070	0.000 ± 0.000	0.000 ± 0.000	0.096 ± 0.030	0.105 ± 0.046
GIN (shuffle)	0.254 ± 0.221	0.289 ± 0.206	0.360 ± 0.224	0.518 ± 0.110	0.254 ± 0.221	0.193 ± 0.167	0.149 ± 0.110	0.149 ± 0.061
NetLSD	0.000 ± 0.000	0.000 ± 0.000	0.088 ± 0.030	0.140 ± 0.040	0.000 ± 0.000	0.000 ± 0.000	0.018 ± 0.015	0.018 ± 0.030

TABLE VII: Accuracy drop ΔAcc under edge perturbations on ENZYMES (mean $\pm$ std over 3 seeds). Lower is better.

	Removal-only (p_r, $p_a = 0$)				Addition-only (p_a, $p_r = 0$)			
Method	0.00	0.05	0.10	0.20	0.00	0.05	0.10	0.20
GIN (no-shuffle)	0.000 ± 0.000	0.006 ± 0.024	0.061 ± 0.013	0.114 ± 0.013	0.000 ± 0.000	0.031 ± 0.013	0.106 ± 0.055	0.172 ± 0.054
GIN (shuffle)	0.178 ± 0.088	0.200 ± 0.102	0.183 ± 0.101	0.200 ± 0.073	0.178 ± 0.088	0.200 ± 0.079	0.192 ± 0.058	0.222 ± 0.067
NetLSD	0.000 ± 0.000	0.008 ± 0.014	-0.011 ± 0.017	0.017 ± 0.036	0.000 ± 0.000	0.008 ± 0.030	0.014 ± 0.034	0.031 ± 0.013

TABLE VIII: Accuracy drop ΔAcc under edge perturbations on IMDB-MULTI (mean $\pm$ std over 3 seeds). Lower is better.

	Removal-only (p_r, $p_a = 0$)				Addition-only (p_a, $p_r = 0$)			
Method	0.00	0.05	0.10	0.20	0.00	0.05	0.10	0.20
GIN (no-shuffle)	0.000 ± 0.000	-0.001 ± 0.011	-0.002 ± 0.023	-0.022 ± 0.043	0.000 ± 0.000	-0.001 ± 0.007	0.001 ± 0.007	-0.003 ± 0.020
GIN (shuffle)	0.009 ± 0.030	0.002 ± 0.034	0.004 ± 0.035	-0.018 ± 0.057	0.009 ± 0.030	0.008 ± 0.028	0.010 ± 0.029	0.010 ± 0.035
NetLSD	0.000 ± 0.000	0.023 ± 0.019	0.031 ± 0.005	0.058 ± 0.033	0.000 ± 0.000	-0.007 ± 0.009	-0.008 ± 0.024	0.018 ± 0.022

D. Efficiency: Runtime

Table IX summarizes the computational cost of each pipeline. For Graph2Vec and NetLSD, we report embedding extraction time and downstream SVM training time. For GIN, we report end-to-end training time (200 epochs), as embeddings are learned jointly with the classifier. We additionally include Optuna hyperparameter tuning overhead for all methods.

TABLE IX: Efficiency metrics (seconds, mean $\pm$ std over 3 seeds).

Method	Dataset	Embed time	Train time	Optuna time
Graph2Vec	IMDB-MULTI	11.23 ± 0.10	1.02 ± 0.23	93.28 ± 2.69
Graph2Vec	ENZYMES	6.66 ± 0.22	0.17 ± 0.06	51.90 ± 0.28
Graph2Vec	MUTAG	1.06 ± 0.25	0.01 ± 0.00	8.50 ± 0.38
NetLSD	IMDB-MULTI	5.38 ± 0.24	1.37 ± 0.20	46.85 ± 0.13
NetLSD	ENZYMES	2.16 ± 0.17	0.25 ± 0.09	19.83 ± 0.59
NetLSD	MUTAG	0.55 ± 0.00	0.01 ± 0.00	4.95 ± 0.77
GIN	IMDB-MULTI	–	76.76 ± 0.56	25.64 ± 6.42
GIN	ENZYMES	–	26.39 ± 1.73	6.48 ± 0.39
GIN	MUTAG	–	11.75 ± 0.73	2.77 ± 0.27

Overall, unsupervised descriptors are computationally lightweight and train the downstream classifier in negligible time, while GIN incurs substantially higher training cost, particularly on larger datasets such as IMDB-MULTI. Notably, hyperparameter tuning dominates the runtime of the SVM-based pipelines, whereas for GIN the main cost stems from full model training. These results highlight a clear accuracy–efficiency trade-off between classical descriptors and supervised GNNs.

VI. Discussion

Our experimental study provides a holistic comparison between classical unsupervised whole-graph descriptors and supervised graph neural networks across four complementary axes: predictive performance, embedding geometry, computational efficiency, and robustness to perturbations. Rather than revealing a universally superior approach, the results highlight clear trade-offs between accuracy, cost, and stability, suggesting that the most suitable method depends strongly on practical constraints.

A. Performance vs. Learning Paradigm

Across all datasets, the supervised GIN model consistently achieves the strongest predictive performance. In particular,

GIN improves accuracy by roughly 6–12 points on MUTAG and more than 20 points on ENZYMES compared to unsupervised alternatives. This behavior is expected, as end-to-end supervision explicitly optimizes representations for the downstream classification objective. Message passing enables the network to learn task-specific structural patterns that fixed descriptors may not capture.

In contrast, Graph2Vec and NetLSD compute task-agnostic embeddings without label information. Although this limits their ability to align embedding geometry with class boundaries, their performance remains competitive on IMDB-MULTI, where simpler global topology appears sufficient for discrimination. This suggests that for problems with coarse or topology-driven distinctions, unsupervised structural descriptors may already capture most of the useful signal without the overhead of neural training.

Overall, these findings confirm a practical rule: supervised GNNs provide the highest peak accuracy, whereas unsupervised embeddings offer strong baselines with substantially lower complexity.

B. Embedding Geometry and Clustering Behavior

Clustering experiments provide additional insight into representation structure. GIN embeddings achieve the highest ARI on MUTAG, indicating that supervision encourages class-aligned organization in the embedding space. This demonstrates that supervised learning not only improves classification metrics but also shapes more semantically meaningful geometry.

However, the observed discrepancy between ARI and Silhouette scores reveals an important nuance. Several methods exhibit high Silhouette values while ARI remains near zero, indicating that compact geometric clusters do not necessarily correspond to true class labels. Consequently, geometric separation alone is insufficient as a proxy for discriminative power, and label-aware metrics such as ARI are more informative for evaluating embedding usefulness in classification settings.

Interestingly, NetLSD often produces smooth and well-separated manifolds despite modest classification performance, suggesting that spectral descriptors capture consistent global structure but lack the adaptive flexibility of learned message passing.

C. Efficiency and Practical Deployment

The computational differences between approaches are substantial. Graph2Vec and NetLSD compute embeddings quickly on CPU and train lightweight SVM classifiers in negligible time, making them attractive for rapid experimentation or resource-constrained environments. In contrast, GIN requires GPU training and significantly longer optimization, especially on larger datasets such as IMDB-MULTI.

These results expose a clear accuracy–efficiency trade-off. While GIN offers superior predictive performance, its computational cost may not be justified for small datasets or low-stakes applications. In practice, unsupervised embeddings can serve as fast baselines or prototypes, whereas supervised GNNs are better suited when maximum accuracy is critical and computational resources are available.

D. Robustness and Stability

The stability analysis reveals complementary robustness properties that are not visible from classification accuracy alone. While supervised GIN achieves the best predictive performance under clean conditions, its behavior under perturbations differs substantially from that of classical structural descriptors, highlighting an additional trade-off between expressiveness and invariance.

a) Spectral invariance of NetLSD.: Across all datasets and perturbation strengths, NetLSD consistently exhibits near-perfect embedding stability, with cosine similarity remaining close to one even under aggressive edge removals or additions. This behavior is expected from its design: the descriptor depends solely on Laplacian eigenvalues and aggregates information over multiple diffusion scales, making it largely insensitive to small local structural edits. As a result, NetLSD shows minimal accuracy degradation on ENZYMES and IMDB-MULTI and only moderate drops on MUTAG. These findings confirm that spectral signatures provide strong structural robustness and deterministic behavior, making them particularly attractive in noisy or evolving graph environments.

b) Corpus sensitivity of Graph2Vec.: Graph2Vec demonstrates intermediate robustness. Although it is less sensitive than GIN to feature noise, its embeddings vary noticeably when the underlying graph corpus is perturbed. This is partly due to its stochastic training objective and reliance on subgraph co-occurrence statistics: even small structural changes alter the distribution of rooted subgraphs and therefore the learned embedding space. Consequently, Graph2Vec exhibits consistent but moderate pipeline drift under refitting, and we do not interpret fixed-classifier accuracy degradation across perturbations as a valid robustness metric for Graph2Vec.

c) Task-adaptivity vs. robustness in GIN.: GIN displays the strongest dependence on the type of perturbation. When only edges are modified, robustness is dataset-dependent: performance remains stable on IMDB-MULTI, yet drops more substantially on MUTAG and ENZYMES. More strikingly, shuffling node attributes consistently leads to larger degradation than edge edits alone. This indicates that GIN relies heavily on the alignment between node features and graph topology, and that disrupting this alignment harms the learned message-passing representations. In other words, the same task-specific adaptivity that enables higher clean accuracy also reduces invariance to structural or feature noise.

d) Dataset effects.: Robustness also varies by domain. IMDB-MULTI is comparatively insensitive for all methods, likely because its classes depend on coarse global connectivity patterns rather than fine-grained local motifs. In contrast, MUTAG and ENZYMES encode chemically or biologically meaningful substructures, where small edge modifications can significantly affect discriminative patterns. This explains the larger observed accuracy drops on these datasets, especially for learned neural embeddings.

e) Implications.: Taken together, these results emphasize that predictive accuracy and stability are not strongly coupled. Highly expressive neural models may achieve superior performance yet be more brittle, whereas simpler spectral descriptors may sacrifice some accuracy while providing strong invariance and determinism. Therefore, robustness considerations should play a central role when selecting whole-graph embedding methods for real-world deployments, particularly in settings where graphs are noisy, incomplete, or subject to temporal evolution.

E. Practical Guidelines

Based on the combined results, the following recommendations emerge:

- Choose **GIN** when labeled data is available and maximizing predictive accuracy is the primary objective.
- Choose **NetLSD** when efficiency, determinism, and structural stability are more important than peak accuracy.
- Use **Graph2Vec** as a lightweight baseline or for rapid prototyping with minimal computational overhead.
- Prefer **spectral descriptors** in applications requiring robustness to structural noise or graph perturbations.
- Evaluate both predictive metrics and embedding geometry, as geometric compactness alone may not imply class discrimination.

F. Threats to Validity and Limitations

Several limitations should be acknowledged. First, unsupervised embeddings were fitted transductively on the full corpus without labels, which may slightly inflate downstream performance compared to a strictly inductive setting. Second, stability comparisons are asymmetric, as Graph2Vec does not provide a stable out-of-sample inference interface. Third, the applied perturbations represent generic stress tests and may not reflect realistic domain-specific noise. Finally, small datasets such as MUTAG exhibit higher variance across splits, and a larger hyperparameter budget could further improve absolute scores.

VII. Conclusion

We conducted a systematic empirical comparison of representative whole-graph embedding methods across multiple datasets and evaluation criteria, including accuracy, clustering structure, efficiency, and robustness.

Our results highlight clear trade-offs: supervised GNNs achieve the highest predictive performance, spectral descriptors offer strong stability and efficiency, and lightweight unsupervised embeddings provide practical low-cost baselines. Importantly, we show that accuracy, geometric separability, and robustness are not strongly coupled.

Overall, selecting an embedding strategy should depend on application constraints rather than accuracy alone. Future work will extend this benchmark to larger and inductive settings.

Reproducibility

To promote transparency and reproducibility, we release the complete experimental pipeline, including data processing, embedding extraction, training, tuning, and evaluation scripts. All results can be reproduced using the provided configuration files and fixed random seeds. The codebase is publicly available at:

https://github.com/ntua-el21090/Graph_Embedding_Eval2025.

References

[1] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, "graph2vec: Learning distributed representations of graphs," in *Proc. IEEE International Conference on Data Mining Workshops (ICDMW)*, 2017.

[2] A. Tsitsulin, D. Mottin, P. Karras, and E. Müller, "Netlsd: Hearing the shape of a graph," in *Proc. ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, 2018.

[3] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *International Conference on Learning Representations (ICLR)*, 2019.

[4] M. Fey and J. E. Lenssen, "Fast graph representation learning with pytorch geometric," in *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[5] C. Morris, N. M. Kriege, F. Bause, W. L. Hamilton, C. Llewellyn, P. Veličković *et al.*, "Tudataset: A collection of benchmark datasets for learning with graphs," *arXiv preprint arXiv:2007.08663*, 2020.