Low-Shot Graph Learning with Topological and Spectral Embeddings

Sai Karthik Navuluru[*]
University of Texas at Dallas
SaiKarthik.Navuluru@utdallas.edu

Surbhi Kumar[*]
University of Texas at Dallas
Surbhi.Kumar@utdallas.edu

Baris Coskunuzer
University of Texas at Dallas
coskunuz@utdallas.edu

Abstract

Deep graph learning has achieved remarkable success, but its reliance on abundant labeled data limits use in many scientific domains, where each label may require costly experiments or simulations. We revisit graph classification in the low-shot regime and ask whether explicit, theory-grounded descriptors can provide a reliable foundation and how they can be combined with modern architectures. We study two families of label-free embeddings: *topological vectors* from persistent homology that capture multiscale connectivity, and *spectral vectors* from the Laplacian density of states that summarize diffusion geometry. To harness their complementary strengths, we also introduce *prototype embeddings*, which project graphs onto class-level prototypes in the joint topological–spectral space, and `STAMP`, a lightweight controller that conditions GNN and GT backbones on these descriptors through layer-wise modulation.

Across ten TU benchmarks and label budgets $K \in \{1, 5, 10, 25, 50\}$, prototype embeddings with simple classifiers consistently outperform strong baselines in the extreme low-label setting, while `STAMP` achieves the best overall performance once $K \geq 10$. Our results demonstrate that explicit structural priors offer a powerful and complementary route to label-efficient graph learning, closing much of the gap to larger deep models without pretraining.

1 Introduction

Graph classification is a fundamental machine learning task with broad applications in drug discovery, materials science, and bioinformatics. In these domains, each labeled graph, for example, a molecule validated in a wet lab, carries a significant cost, so labeled data are scarce. This scarcity creates a bottleneck: state-of-the-art Graph Neural Networks (GNNs) and Graph Transformers (GTs) benefit from large datasets, yet many classical benchmarks have only hundreds of graphs in total [1]. Even as larger collections appear through OGB [2], rare classes and limited label budgets remain common in practice. Designing graph models that perform reliably with only a few dozen labeled examples is therefore both scientifically challenging and practically important.

The difficulty in low-shot graph learning comes not only from limited supervision but also from architectural vulnerabilities in deep models. With few labels, gradients update parameters sparsely, which leads to overfitting on narrow regions of the feature space. Over-smoothing and misaligned readouts further degrade quality, causing node embeddings to collapse and graph-level signals to weaken. As a result, sophisticated GNNs and GTs can underperform well-engineered classical approaches. Prior work shows that aggregations of many interpretable graph features, subtree-kernel methods, and spectral diffusion signatures achieve strong accuracy on small benchmarks such as MUTAG and ENZYMES when coupled with robust shallow classifiers [3–5]. These observations

[*] Equal contribution.

Navuluru et al., Low-Shot Graph Learning with Topological and Spectral Embeddings. *Proceedings of the Fourth Learning on Graphs Conference (LoG 2025)*, PMLR 269, Hybrid Event, December 10–12, 2025.

suggest a broader lesson: in data-scarce regimes, stable structural descriptors grounded in theory can rival or surpass modern deep architectures.

Motivated by this, we ask: *Can direct, theory-grounded graph embeddings provide a reliable foundation for low-shot classification, and how can they be combined with modern architectures?* We study two complementary families of label-free descriptors: *topological embeddings* from persistent homology, which capture multiscale connectivity in stable, permutation-invariant vectors [6–8], and *spectral embeddings* based on Laplacian density of states, which summarize diffusion and volume growth through efficiently approximated eigenvalue distributions [5, 9, 10]. Both are efficient, interpretable, and stable under perturbations, making them well-suited for low-data settings. In addition, we introduce *prototype embeddings* that combine these descriptors by projecting each graph onto class-level prototypes in the joint topological–spectral space, yielding a compact similarity representation that excels in extreme low-label regimes.

Building on these descriptors, we introduce a compact *hybrid* model that augments GNNs and GTs with topological and spectral priors. The design integrates nonparametric embeddings by simple concatenation and conditioning, improves label efficiency, and does not rely on pretraining.

Our Contributions.

- We provide a systematic evaluation of topological and spectral embeddings for *low-shot graph classification*, demonstrating strong performance without learned parameters.
- We introduce *prototype embeddings* that fuse topological and spectral descriptors via class-level prototypes, delivering the strongest performance at very small label budgets.
- We propose `STAMP`, a hybrid model that integrates spectral–topological descriptors into GNNs and GTs, which reduces overfitting and improves sample efficiency.
- We analyze when and why these embeddings help, showing consistent improvements across diverse datasets and label budgets with $K \in \{1, 5, 10, 25, 50\}$.

2 Background

2.1 Related Work

Few-Shot Learning for Graph Classification. Meta-learning and metric-based pipelines dominate few-shot graph classification, typically combining episodic training with task-specific graph encoders. Representative approaches enrich message passing with structural priors or hierarchical task relations, for example, structure-enhanced meta-learning [11], FAITH with hierarchical task graphs [12], and prototype-based methods that learn distance metrics over graph representations [13]. These strategies can transfer across tasks when many related episodes are available, yet their effectiveness hinges on learned encoders and careful episodic design.

A complementary literature explores stronger encoders, including Graph Transformers that benefit from scale or pretraining [14–16]. However, under scarce supervision, these models often overfit or require large unlabeled corpora, and empirical audits on small benchmarks show that sophisticated GNNs may fail to consistently outperform well-tuned classical baselines [17]. We take a different route. We start from label-free, theory-grounded graph embeddings and either use them directly or inject them as compact tokens into a lightweight Transformer. This yields strong inductive biases without reliance on heavy pretraining while remaining compatible with standard few-shot protocols.

TDA in Graph Learning. Topological Data Analysis provides stable, permutation-invariant summaries of multiscale connectivity, commonly via persistent homology (PH) diagrams and their vectorizations. On graphs, PH has been exploited through learnable filtrations and differentiable readouts for graph classification [18, 19], neural layers operating directly on persistence diagrams with graph-specific signatures such as HKS [8], WL-style augmentations that inject cycle information for improved expressivity [20], and architectures that reweight message passing with PH-derived signals [21]. Recent work also injects global topological invariants into pooling layers [22], uses multiparameter persistence in the drug discovery problem [23], learns fast surrogates for extended PH on graphs [24], and captures localized structure with persistent local homology [25]. While these studies establish the utility of TDA for graph learning, most do not target the low-shot fixed split setting or hybridize PH descriptors with compact Transformers. We close this gap by evaluating PH vectors explicitly under few-label budgets and by integrating them as inductive-bias tokens within a lightweight Transformer.

Spectral Methods in Graph Learning. Spectral descriptors summarize global structure through functionals of the graph Laplacian [26, 27]. NETLSD constructs permutation and size invariant signatures from heat and wave kernel traces [5]. Density-of-states (DoS) methods approximate the full eigenvalue distribution efficiently via polynomial approximation and stochastic trace estimation, yielding compact graph fingerprints that scale to large graphs [9, 10, 28]. Beyond scalar spectra, spectral graph wavelets and their scattering extensions provide multiscale, stable features that align with convolutional architectures and have proven effective for graph analysis [29–32]. Despite this evidence, spectral summaries are rarely positioned as backbones for *low-shot, fixed-split* classification. We treat DoS vectors as direct graph embeddings that are competitive on their own and complementary to persistent-homology summaries, and we show that combining both as tokens for a small Transformer yields data-efficient hybrids.

2.2 Persistent Homology

Persistent Homology (PH) is a fundamental tool in TDA that captures multiscale structural features such as connected components, cycles, and higher-order cavities that persist across resolutions [33]. Initially developed for point clouds, PH has since been extended to graphs, images, and other modalities [34]. Within graph learning, PH provides latent descriptors that complement message-passing networks by exposing higher-order connectivity patterns beyond local neighborhoods.

A PH pipeline has three steps: *filtration*, *diagram construction*, and *vectorization*. Filtration builds a nested sequence of subgraphs or simplicial complexes according to a chosen function $f : \mathcal{V} \to \mathbb{R}$ (e.g., degree, centrality, domain-specific node attributes). For weighted graphs, edge weights naturally define the order (Figure 1). Each filtration yields a sequence $\widehat{\mathcal{G}}^1 \subseteq \widehat{\mathcal{G}}^2 \subseteq \cdots \subseteq \widehat{\mathcal{G}}^N$, where topological features appear (birth) and disappear (death). These events are summarized as points (b_σ, d_σ) in persistence diagrams $\mathrm{PD}_k(\mathcal{G})$ indexed by homology dimension k. Since diagrams are multisets in $\mathbb{R}^2$, they must be vectorized for compatibility with learning pipelines. Common choices include persistence images, landscapes, silhouettes, and Betti curves [35].

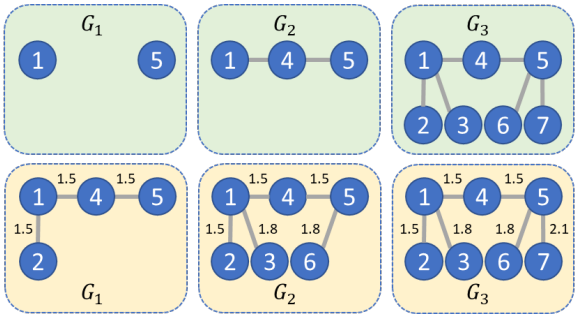


Figure 1: Graph Filtration. For $\mathcal{G} = \mathcal{G}_3$ in both examples, the top figure illustrates a *superlevel filtration using the node degree function* with thresholds $3 > 2 > 1$, where nodes of degree 3 are activated first, followed by those of lower degrees. Similarly, the bottom figure illustrates a *sublevel filtration based on edge weights* with thresholds $1.5 < 1.8 < 2.1$.

PH is appealing in low-data regimes: its descriptors are permutation-invariant, computationally efficient, and come with stability guarantees [6]. Moreover, unlike deep GNNs that require task-specific training, PH embeddings provide robust, theory-grounded priors that can be directly fed into classifiers or combined with neural architectures. In this work we leverage PH vectors as a strong baseline and as a complementary signal for hybrid models.

2.3 Density of States for Graphs

The *density of states* (DoS) is a classical spectral descriptor that summarizes the eigenvalue distribution of a graph Laplacian [27]. Given a normalized Laplacian L of graph $G = (\mathcal{V}, \mathcal{E})$ with eigenvalues $\{\lambda_i\}_{i=1}^{|\mathcal{V}|}$, the DoS is defined as $\mathrm{DoS}(\lambda) = \frac{1}{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{V}|} \delta(\lambda - \lambda_i)$, where $\delta(\cdot)$ is the Dirac delta. Intuitively, DoS encodes how eigenvalues concentrate across $[0, 2]$, reflecting global connectivity, diffusion, and bottleneck structures. In practice, computing all eigenvalues is infeasible for large graphs. Instead, polynomial approximation techniques [10] and stochastic trace estimation yield efficient histograms over fixed bins in $[0, 2]$ [9]. The resulting vector is compact, permutation-invariant, and stable to small perturbations, while encoding graph-theoretic properties such as connectivity, expansion, and clustering tendencies [5, 36].

As with PH, DoS embeddings are label-free, fast to compute, and effective in low-shot settings where deep models overfit. They provide a complementary view: while PH emphasizes topological persistence of features, DoS captures spectral smoothness and diffusion behavior. We exploit both as standalone descriptors and as auxiliary tokens in our hybrid Graph Transformer.

2.4 Problem setting: low-shot fixed split

To clarify our evaluation protocol, we distinguish our *low-shot fixed-split* setting from the more common *episodic few-shot learning (FSL)* framework. Episodic FSL trains and evaluates over many support–query episodes sampled from a distribution of tasks, with the goal of rapid adaptation to new classes. In contrast, our focus is on a single fixed label space, where the challenge lies in training with only K labeled graphs per class and evaluating on a held-out test set. This distinction reflects realistic deployment scenarios in graph learning, where classes remain fixed but labeled data is scarce. Table 1 summarizes the key differences between these two paradigms.

Table 1: Comparison of episodic few-shot learning (FSL) and our low-shot fixed-split setting.

	Episodic FSL (meta-learning)	**Low-shot fixed split (ours)**
Goal	Learn to adapt quickly to *new tasks/classes*.	Achieve high accuracy with *few labels* for a fixed task (same classes).
Unit of training	Thousands of *episodes*, each a small classification problem (support+query).	One fixed training set with K labeled graphs per class.
Support set	Used to adapt within each episode.	Equivalent to the K training examples per class.
Query set	Required: evaluates adaptation in each episode.	Not used: evaluation is on a fixed held-out test set.
Dataset requirement	Many classes (to sample diverse tasks).	Works with small-class datasets common in graph benchmarks.
Evaluation	Average accuracy across many test episodes with unseen classes.	Accuracy on a fixed test set; report results for $K \in \{1, 5, 10, 25, 50\}$.

3 Methodology

Our objective is to assess the effectiveness of direct graph-level embeddings that are computed without learned weights as building blocks for graph classification in low-label settings. We focus on three complementary families: (i) topological embeddings derived from persistent homology, which capture multiscale connectivity; (ii) spectral embeddings based on density-of-states vectors, which summarize global diffusion geometry; and (iii) prototype-based embeddings induced from (i) and (ii). We benchmark these approaches against strong parametric baselines, including Graph Neural Networks (GNNs) and Graph Transformers (GTs), and further introduce a compact hybrid model that integrates non-parametric descriptors with GNNs and GTs. This design aims to combine the inductive biases and robustness of explicit embeddings with the adaptability of deep architectures, yielding a data-efficient and competitive solution for low-shot graph classification.

3.1 Direct Embeddings

Direct embeddings provide a way to map entire graphs into fixed-length vectors without relying on iterative message passing or deep architectures. Instead, they directly summarize global graph information through structural invariants, spectral distributions, or distances to learned prototypes. In this work, we consider three complementary strategies: topological embeddings, which capture connectivity patterns through Betti-based descriptors; spectral embeddings, which summarize Laplacian eigenvalue distributions via Density of States histograms and kernel density estimates; and prototype-based representations, which measure similarity to class-level prototypes in feature space. Each of these embeddings encodes distinct aspects of graph structure, and together they provide a versatile feature space for downstream classification tasks.

Topological Embeddings. Topological embeddings are obtained using Betti vectorizations derived from both degree sublevel and diffusion-based (HKS) filtrations, yielding fixed-length, graph-level representations by tracking the evolution of connected components β_0 and cycles β_1 across thresholds. In the degree-based approach, thresholds are sampled between the minimum and maximum node degree, and Betti numbers are computed on the induced subgraphs, producing a 2T-dimensional descriptor with stability ensured for empty or constant-degree graphs. In the HKS-based approach, nodewise heat kernel signatures are computed from Laplacian eigenpairs at multiple diffusion times, and thresholding these values yields Betti vectors that capture multiscale structural patterns. Together, the two filtrations provide complementary information, with degree-based vectors emphasizing coarse connectivity and HKS-based vectors capturing diffusion-aware structure.

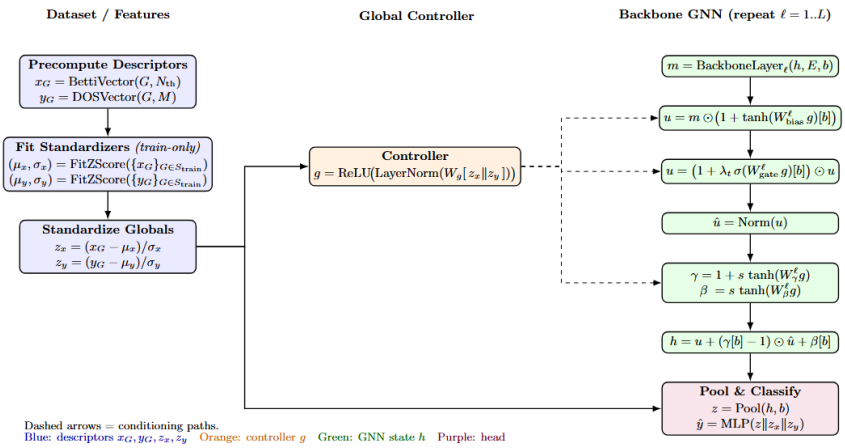


Figure 2: `STAMP flowchart.` Left: dataset-level descriptors are precomputed and standardized to (z_x, z_y) using train-only statistics. Middle: a global controller g is produced. Right: g modulates every GNN layer via bias, gating, and affine normalization (γ, β); the pooled graph embedding is concatenated with (z_x, z_y) and fed to an MLP to predict $\hat{y}$. Dashed arrows denote conditioning paths; colors indicate modules.

Spectral Embeddings. Spectral embeddings are derived from the eigenvalue distribution of the normalized Laplacian, which encodes fundamental properties of graph connectivity. The distribution is summarized through the Density of States (DOS), providing a global spectral fingerprint of the graph. Two variants are considered: (i) histogram-based DOS, where eigenvalues are binned into fixed intervals over $[0, 2]$ to produce a normalized frequency vector, and (ii) KDE-based DOS, where eigenvalues are smoothed using kernel density estimation with boundary reflection, yielding a continuous approximation of the spectral density. Both representations are normalized to form probability vectors, ensuring comparability across graphs of different sizes. Histogram embeddings emphasize coarse spectral structure, while KDE embeddings capture smoother and more fine-grained variations in the spectrum, making them robust descriptors for graph-level classification.

Prototype-based Embeddings. Prototype-based representations encode graphs through their similarity to class-level prototypes in a chosen feature space. To fully leverage direct embeddings above, we aggregate them via prototype construction in both topological and spectral domains: Betti vectors from degree and HKS filtrations, and DOS descriptors from histogram and KDE estimates. For each class, a prototype is obtained by averaging the embeddings of its training graphs. A graph is then re-embedded by measuring its distance to all prototypes, using either Euclidean or cosine similarity. This yields a compact, discriminative coordinate system where positions reflect similarity to class prototypes rather than raw descriptors. In doing so, prototype-based embeddings unify the stability of handcrafted topological and spectral features with the discriminative power of metric learning.

3.2 STAMP — Spectral-Topological Augmented Message Passing*

We propose `STAMP` (Figure 2), a plug-and-play global-feature controller for graph neural networks (GNNs), supporting two interchangeable backbones: (i) `STAMPGPS`, which leverages GPSConv layers combining local message passing with global attention, and (ii) `STAMPGCN`, a lightweight variant based on standard GCN layers.

Each graph G is first augmented with global structural descriptors: the Betti vector $x_G \in \mathbb{R}^{2N_{\text{th}}}$, and the Laplacian DOS vector $y_G \in \mathbb{R}^M$. These are standardized as $z_x = \dfrac{x_G - \mu_x}{\sigma_x}, \quad z_y = \dfrac{y_G - \mu_y}{\sigma_y}.$

A shared controller encodes these standardized features into a global context vector:

$$g = \text{ReLU}\Big(\text{LayerNorm}\big(W_g[z_x \,\|\, z_y]\big)\Big).$$

At each layer ℓ, this context modulates the message-passing outputs through learned scaling and bias terms: $h^{(\ell)} = u^{(\ell)} + (\gamma^\ell[b] - 1) \odot \hat{u}^{(\ell)} + \beta^\ell[b],$ where $u^{(\ell)}$ is the backbone output and $(\gamma^\ell, \beta^\ell)$ are layer-specific modulation parameters derived from g.

After L layers, node embeddings are pooled: $z = \text{Pool}\big(h^{(L)}, b\big),$ and combined with the standardized global features for classification: $o = \text{MLP}\big([z \,\|\, z_x \,\|\, z_y]\big).$

Detailed descriptions of the training and forward-pass algorithms, with pseudocode, are provided in Appendix C.

3.3 Stability of Topological and Spectral Embeddings.

In graph learning, robustness to small perturbations in the graph structure is critical, particularly in real-world settings where data may be noisy, or incomplete. We utilize topological and spectral descriptors that are inherently stable: small changes to the graph, such as edge insertions or deletions induce only bounded changes in the extracted features. This stability ensures that the learned representations are resilient to noise, which is essential for reliable graph classification. We formalize these stability properties below. First, we state the stability of our topological descriptors.

Theorem 3.1 (Stability of Topological Descriptors)**.** *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph and let $f, g : \mathcal{V} \to \mathbb{R}$ be two node filtration functions on $\mathcal{G}$. Then, for $k \geq 0$, we have $\|\vec{\beta}_k(\mathcal{G}, f) - \vec{\beta}_k(\mathcal{G}, g)\|_1 \leq C_k \cdot \|f - g\|_1$ where $\vec{\beta}_k(\mathcal{G}, f)$ represents the Betti vector corresponding to sublevel filtration with respect to f.*

Next theorem is on the stability of our spectral descriptors.

Theorem 3.2 (Stability of Spectral Descriptors)**.** *Let $\mathcal{G}$ and $\mathcal{G}'$ be two graphs differing by at most k edge modifications (insertions or deletions). Then, the Wasserstein distance between the spectral descriptors ψ and ψ' is bounded by Ck/n, where $n = |\mathcal{V}|$ and the constant C depends on the eigenvalue distribution.*

The proofs of the theorems are given in Appendix A.

4 Experiments

Datasets. To assess the effectiveness of our models across diverse settings, we utilize ten graph classification benchmark datasets from the TU Collection [1]: (i) molecular and biological graphs from BZR, COX2, MUTAG, ENZYMES, and PROTEINS [37], and (ii) social graphs, including IMDB-Binary, IMDB-Multi, COLLAB, REDDIT-Binary, and REDDIT-Multi-5K [38].

Low-shot evaluation protocol. The data were split into three disjoint partitions: a **training pool**, a **fixed validation set** with 10 samples per class, and a **fixed test set** comprising approximately 20% of the dataset. These splits were generated using a global random seed and remained fixed across all runs to ensure reproducibility. For each $K \in \{1, 5, 10, 25, 50\}$, we sampled K labeled support examples per class from the training pool using stratified random sampling. Each experiment was repeated across 20 random seeds, and for each model, we reported the mean test accuracy and the standard deviation across these seeds. Results, along with per-seed accuracies, were logged in JSON format for reproducibility and post-hoc statistical analysis.

Table 2: Characteristics of the benchmark graph classification datasets.

Datasets	#Graphs	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	Classes
BZR	405	35.75	38.36	2
COX2	467	41.22	43.45	2
MUTAG	188	17.93	19.79	2
PROTEINS	1113	39.06	72.82	2
ENZYMES	600	32.63	62.14	6
IMDB-B	1000	19.77	96.53	2
IMDB-M	1500	13.00	65.94	3
REDDIT-B	2000	429.63	497.75	2
REDDIT-5K	5000	508.52	594.87	5
COLLAB	5000	74.49	2457.78	3

Hyperparameters for Direct Embeddings. For topological embeddings, the number of thresholds for Betti vectorization was fixed to $T = 10$, while diffusion-based features were computed using Heat Kernel Signatures (HKS) at three diffusion times $\{0.1, 0.5, 1.0\}$. Spectral embeddings were parameterized by 128 bins for DOS histograms and by $m = 128$ grid points with a Gaussian kernel of bandwidth 0.01 for DOS-KDE. Prototype-based representations were constructed using both Euclidean (l_2) and cosine distances. The downstream classifier was a multilayer perceptron trained with a maximum of 5000 iterations, with hyperparameters selected from a fixed grid comprising hidden layer sizes $(32), (64), (64, 32), (128), (128, 64)$, L_2 regularization penalties $\alpha \in \{10^{-4}, 10^{-3}\}$, and learning rates $\eta \in \{10^{-3}, 10^{-2}\}$.

Hyperparameters for STAMP. For each dataset, we performed standardized preprocessing to ensure consistent inputs across models. When node-level attributes were absent or empty, we applied one-hot degree encodings capped at 100 to create structural node features. Labels were converted to zero-indexed format for datasets where class labels were originally one-indexed.

Table 3: Low-shot results. Accuracy for $K = 1, 5, 10, 25, 50$ across topological (Topo), spectral (DOS), prototype-based, DL, and hybrid models. Column-wise best performance is given in blue, the second in purple, and the third in green. The final columns report the average rank and average gap from the best performance.

1-shot Results

Model	BZR	COX2	MUTAG	PROTEINS	ENZYMES	IMDB-B	IMDB-M	REDDIT-B	REDDIT-5K	COLLAB	Av.Rank	Av.Gap
Topo-degree	53.33±16.71	48.49±15.14	62.11±14.08	**59.17±10.86**	17.50±3.60	**52.90±6.45**	**36.00±3.32**	**62.62±8.27**	28.92±5.52	42.49±10.53	**6.5**	**4.0**
Topo-HKS	54.14±13.55	51.83±18.10	**66.32±15.69**	56.41±9.40	17.04±3.20	48.07±5.35	34.28±3.08	**60.26±10.50**	**30.04±5.43**	**44.41±9.54**	7.9	**4.1**
DOS-hist	**59.75±18.33**	46.08±26.60	60.26±12.78	52.83±8.95	17.54±3.74	50.20±5.09	35.10±3.66	52.69±5.67	27.05±4.69	**45.75±12.32**	9.9	5.7
DOS-KDE	57.22±16.48	52.04±16.77	53.29±17.07	53.68±10.38	19.42±3.69	50.52±4.97	35.40±3.19	52.75±5.75	27.29±5.47	43.81±8.29	8.8	5.9
Proto-Euc.	56.05±22.58	**60.65±25.00**	50.79±17.87	52.71±9.84	17.17±3.14	50.82±3.17	34.92±2.74	52.50±6.44	**30.84±4.03**	38.97±11.79	9.9	5.9
Proto-cosine	55.00±15.21	48.17±16.61	**68.42±11.13**	**58.59±10.75**	19.13±3.58	52.83±6.86	**35.98±3.70**	**64.75±6.56**	**32.52±6.30**	**48.26±6.14**	**4.4**	**2.0**
GCN	**59.63±14.42**	51.56±17.79	56.45±10.28	**58.25±6.19**	19.58±4.58	51.35±5.17	**36.10±3.65**	51.38±4.36	29.85±3.68	36.83±9.68	**7.6**	5.3
GAT	53.77±23.14	52.58±19.48	58.82±11.54	58.18±5.76	19.29±3.67	51.80±5.55	34.35±4.34	54.29±10.67	26.50±5.63	38.54±12.58	8.6	5.6
GSAGE	56.23±17.35	**52.63±15.43**	56.71±9.72	58.05±6.73	**21.00±3.29**	51.57±5.32	33.87±5.04	53.01±8.05	25.26±5.06	37.81±12.26	8.9	5.8
GIN	**58.46±14.56**	49.57±13.46	**66.05±13.54**	54.57±6.69	19.54±3.89	51.08±5.39	35.63±3.32	53.68±10.47	27.33±5.61	35.73±12.77	8.4	5.2
GPS	53.58±22.76	49.89±21.54	65.26±14.50	56.95±7.74	**20.46±3.93**	**53.05±5.95**	35.00±3.97	53.20±7.69	27.68±7.53	38.14±13.58	7.5	5.1
GCL	51.85±18.15	51.24±14.74	59.74±13.45	54.84±6.80	18.79±4.15	51.73±4.99	35.17±3.78	51.41±8.93	24.44±5.36	38.49±7.69	11.2	6.6
SGFormer	49.01±24.29	43.71±28.13	60.00±17.15	56.35±7.20	18.92±3.59	51.87±5.51	35.55±4.36	51.34±4.63	25.14±6.46	38.42±11.83	11.3	7.4
Polynormer	46.17±25.81	**52.74±17.65**	64.21±10.86	56.12±7.03	20.42±4.71	52.20±5.14	35.25±3.71	51.48±4.86	26.05±5.69	41.12±13.96	8.1	5.8
STAMP-GCN	53.27±18.67	49.41±20.39	52.24±16.19	53.72±10.94	19.83±4.12	**53.15±6.72**	34.53±4.17	54.02±9.49	29.37±6.31	38.70±9.98	9.2	6.6
STAMP-GPS	54.01±17.87	50.59±18.83	49.61±16.08	55.87±8.98	**21.17±4.86**	52.40±6.71	34.63±4.47	54.40±10.96	29.52±6.52	39.35±9.86	7.8	6.2

5-shot Results

Model	BZR	COX2	MUTAG	PROTEINS	ENZYMES	IMDB-B	IMDB-M	REDDIT-B	REDDIT-5K	COLLAB	Av.Rank	Av.Gap
Topo-degree	**62.16±7.20**	52.63±12.37	77.37±8.04	58.21±5.74	20.92±3.98	57.35±7.11	34.72±3.05	**70.14±8.44**	37.86±3.65	49.05±6.55	6.5	3.2
Topo-HKS	57.65±7.24	51.88±9.28	**78.55±9.39**	61.10±6.70	19.13±2.90	52.80±6.33	34.20±3.27	**67.16±6.09**	37.62±2.95	48.35±4.79	9.3	4.4
DOS-hist	58.40±10.97	52.20±10.50	66.97±8.55	55.43±6.74	18.75±3.01	57.78±5.64	**37.12±4.32**	60.49±6.14	33.48±2.26	52.24±6.50	8.7	5.9
DOS-KDE	55.68±14.01	51.88±8.59	66.97±10.34	57.15±6.11	20.38±4.44	55.38±6.06	36.60±4.47	63.64±4.48	34.71±2.56	53.94±3.45	8.6	5.6
Proto-Euc.	54.07±13.02	52.96±12.97	71.84±11.35	54.51±8.90	20.83±2.58	56.23±5.90	34.87±2.48	55.83±10.14	38.76±3.83	52.73±5.22	9.5	5.9
Proto-cosine	55.06±8.26	51.02±8.90	**80.26±7.74**	**61.19±7.19**	21.21±3.60	57.03±5.20	36.22±4.35	64.98±7.85	**40.42±3.55**	**54.84±5.11**	**5.7**	**3.0**
GCN	60.37±12.83	51.67±11.16	67.37±7.18	56.66±6.55	19.63±4.31	**58.42±4.85**	36.42±4.11	53.71±4.93	37.51±2.34	**54.76±4.94**	7.7	5.5
GAT	52.90±20.03	47.96±21.26	61.32±10.23	53.79±9.28	18.83±4.33	57.20±5.84	**36.62±4.42**	56.76±6.89	33.71±3.91	**54.05±5.10**	11.5	7.9
GSAGE	57.35±11.16	51.94±12.96	65.00±5.65	55.61±6.31	20.21±4.31	57.75±5.41	**37.12±4.64**	56.06±5.41	34.60±3.06	53.82±5.62	8.3	6.2
GIN	60.25±11.57	**53.12±8.16**	**76.58±4.99**	55.09±6.48	22.04±2.38	57.82±4.63	35.48±3.84	61.01±6.36	34.08±2.97	52.09±7.48	7.4	4.4
GPS	53.21±21.28	47.58±17.43	**83.68±7.61**	57.00±7.52	**24.00±3.54**	56.12±6.45	36.75±3.67	52.19±6.33	30.41±9.17	53.12±6.10	9.4	5.8
GCL	60.62±11.85	**55.43±11.41**	76.97±7.30	54.06±5.92	21.04±3.58	56.80±5.08	35.70±4.06	53.67±7.05	32.22±4.03	46.14±5.69	9.4	5.9
SGFormer	51.36±24.64	**53.87±23.57**	65.00±11.62	54.04±9.43	20.17±3.30	**58.83±4.96**	**36.90±4.73**	51.99±5.13	28.21±5.73	54.10±6.16	9.8	7.7
Polynormer	52.22±22.62	48.39±18.36	68.68±1.42	54.48±9.03	20.29±3.43	57.38±5.88	35.52±4.79	52.40±3.61	31.57±7.27	49.05±9.94	12.3	8.2
STAMP-GCN	**61.48±9.76**	48.39±11.03	76.05±10.69	**63.70±6.86**	**22.25±3.58**	**57.98±4.62**	36.18±4.08	**68.99±9.60**	**38.87±4.10**	51.61±6.35	**5.4**	**2.6**
STAMP-GPS	**60.68±16.80**	50.54±15.36	76.05±9.95	**63.32±4.77**	**25.63±4.58**	57.48±5.70	35.62±3.73	64.95±9.87	**38.85±4.22**	50.42±7.41	**6.3**	**2.8**

10-shot Results

Model	BZR	COX2	MUTAG	PROTEINS	ENZYMES	IMDB-B	IMDB-M	REDDIT-B	REDDIT-5K	COLLAB	Av.Rank	Av.Gap
Topo-degree	**67.22±6.12**	55.38±9.67	**82.24±6.33**	61.12±4.52	22.92±4.48	58.83±6.54	36.00±3.32	**76.51±6.96**	40.87±2.75	51.64±4.19	6.7	**3.0**
Topo-HKS	58.46±7.05	54.46±5.64	81.71±6.03	61.77±5.86	21.12±2.53	55.83±4.14	34.58±3.03	68.86±5.64	39.45±3.00	48.56±3.83	10.0	5.8
DOS-hist	59.38±7.87	53.71±8.97	69.08±7.25	60.72±5.13	20.04±3.15	59.55±5.66	37.75±5.09	62.31±5.03	34.54±1.72	54.35±4.51	10.4	7.2
DOS-KDE	58.09±9.82	53.23±7.99	71.97±11.36	58.54±5.21	22.42±3.70	59.45±4.58	38.13±3.12	66.83±4.26	36.29±2.41	56.64±4.40	9.1	6.2
Proto-Euc.	57.28±7.05	49.84±8.58	76.71±8.78	53.81±8.52	22.46±3.09	58.07±4.07	36.07±3.59	67.79±9.63	39.57±2.86	53.36±4.04	11.3	6.8
Proto-cosine	58.89±8.46	51.67±9.01	**82.89±4.52**	**62.35±4.83**	22.21±4.03	**60.30±5.20**	38.10±3.51	65.59±6.88	**40.92±2.76**	56.87±4.35	**6.2**	4.3
GCN	61.17±12.77	52.85±11.98	67.76±5.76	56.59±6.29	21.08±2.85	**60.20±5.42**	**39.33±4.21**	57.01±4.05	38.49±1.71	**59.35±4.67**	8.3	6.9
GAT	58.46±16.86	51.24±19.61	62.37±7.94	54.08±7.10	19.38±3.00	59.58±3.90	**39.18±4.19**	59.16±4.44	34.63±4.94	**58.23±5.11**	10.6	8.7
GSAGE	60.12±11.32	55.27±9.54	66.45±6.16	57.09±4.66	21.50±3.03	58.93±4.78	38.93±4.66	59.15±3.87	38.22±2.30	57.75±4.80	9.0	7.0
GIN	62.53±11.11	**60.91±6.85**	77.76±5.09	56.93±4.13	**25.63±2.79**	58.53±5.55	37.73±3.66	63.45±3.27	35.67±2.57	**58.42±4.20**	7.9	4.6
GPS	52.10±24.11	54.78±18.24	**83.42±12.40**	60.49±6.07	**28.58±4.09**	59.18±4.75	37.82±3.65	56.61±7.84	39.82±2.51	56.79±6.03	7.7	5.4
GCL	64.63±7.32	**60.97±11.64**	78.42±3.96	56.30±4.61	23.79±4.34	58.88±5.66	36.72±3.77	60.50±4.46	36.57±2.93	50.03±4.74	9.1	5.6
SGFormer	43.09±23.11	56.24±25.17	67.76±8.48	52.09±7.49	22.25±2.50	**60.20±4.67**	38.78±3.33	53.90±5.38	38.02±3.36	57.50±5.51	9.9	9.3
Polynormer	47.72±22.96	47.63±18.07	68.82±1.51	57.15±5.70	22.71±3.57	59.40±4.37	**39.00±4.36**	55.93±5.51	37.40±2.89	55.20±5.04	10.6	9.2
STAMP-GCN	**64.88±8.26**	57.90±9.71	79.87±10.70	**62.17±7.63**	25.04±3.38	**59.88±3.80**	37.97±4.34	**76.46±6.84**	**42.67±2.52**	56.44±4.89	**4.5**	**2.0**
STAMP-GPS	**65.56±7.81**	**59.41±14.50**	81.84±8.02	**64.39±7.00**	**29.00±3.32**	58.68±4.64	38.22±3.73	**70.35±8.57**	**42.11±2.83**	54.55±6.46	**4.6**	**1.9**

25-shot Results

Model	BZR	COX2	MUTAG	PROTEINS	ENZYMES	IMDB-B	IMDB-M	REDDIT-B	REDDIT-5K	COLLAB	Av.Rank	Av.Gap
Topo-degree	71.54±5.04	55.38±7.73	**84.87±3.62**	**64.22±3.38**	23.58±3.14	61.73±4.31	40.25±2.50	**81.24±2.98**	**44.74±1.42**	55.73±2.99	7.7	4.7
Topo-HKS	64.38±5.98	54.89±6.99	81.58±5.77	63.77±3.00	22.92±3.46	57.30±4.36	36.93±3.27	73.98±3.19	42.98±1.78	52.17±2.02	10.0	7.9
DOS-hist	64.20±7.48	60.86±8.06	70.79±6.55	61.86±3.23	22.75±3.43	62.85±4.58	40.83±3.34	67.60±3.01	37.32±1.14	58.69±2.16	9.7	8.2
DOS-KDE	61.36±5.83	56.99±6.52	74.08±6.57	59.53±4.40	24.83±3.15	62.80±4.64	40.77±3.06	72.41±3.03	38.09±1.61	60.97±2.23	9.9	7.8
Proto-Euc.	60.31±5.41	56.40±7.79	79.61±7.10	60.99±4.44	23.92±3.86	62.50±4.87	39.43±2.98	74.64±4.42	42.63±3.02	58.06±2.82	10.0	7.1
Proto-cosine	62.04±5.49	58.23±8.47	81.84±4.47	59.89±3.75	25.00±3.25	63.90±4.65	39.92±3.25	70.40±6.07	41.05±2.39	59.10±2.12	8.9	6.8
GCN	68.77±6.28	53.01±10.30	65.53±5.12	59.84±2.67	23.54±2.81	**67.10±4.16**	**41.53±3.99**	60.39±4.37	41.12±2.13	**63.86±3.24**	8.0	8.5
GAT	53.89±13.32	47.31±17.13	60.92±7.22	57.38±5.84	22.08±2.08	**64.93±4.02**	**41.28±3.94**	58.48±3.02	36.65±3.99	61.36±3.69	12.2	12.6
GSAGE	69.38±5.85	59.09±11.80	64.87±5.72	59.04±3.95	23.62±2.18	64.43±3.75	**41.15±4.19**	60.34±3.15	40.84±1.63	61.95±3.32	8.6	8.5
GIN	**73.64±5.92**	**63.66±6.11**	79.47±4.21	60.74±4.13	28.33±3.43	63.13±3.82	40.60±3.80	66.11±3.66	37.51±1.80	**62.35±3.36**	7.1	5.4
GPS	66.17±7.91	58.98±10.87	**85.53±4.60**	63.52±5.13	**34.40±3.91**	63.68±4.65	40.97±4.29	71.12±3.21	42.91±3.06	60.82±3.15	**5.5**	**4.2**
GCL	71.11±7.73	**66.24±8.99**	80.79±4.41	60.31±3.68	28.46±4.02	62.60±4.97	38.82±3.88	66.31±2.77	38.98±2.05	59.63±3.60	8.6	5.7
SGFormer	58.15±15.69	51.99±14.48	65.39±10.92	59.82±6.99	22.54±2.98	**66.48±3.63**	40.67±3.50	59.29±3.72	40.31±2.96	**62.49±3.29**	10.8	10.3
Polynormer	62.16±10.79	50.59±13.29	68.29±1.95	60.52±4.34	25.08±3.32	64.55±3.55	40.82±3.91	64.65±4.25	38.20±1.99	60.89±3.96	9.6	9.4
STAMP-GCN	**72.78±3.95**	58.87±7.94	80.79±6.45	**66.46±5.93**	**29.92±2.61**	63.25±3.61	40.17±4.16	**83.56±2.63**	**45.71±1.94**	61.06±3.55	**5.0**	**2.7**
STAMP-GPS	**72.96±5.57**	**61.29±8.25**	**83.42±5.46**	**67.83±5.98**	**29.50±3.58**	62.70±3.47	40.00±4.36	**84.01±2.86**	**45.67±1.71**	61.30±4.46	**4.5**	**2.1**

50-shot Results

Model	BZR	COX2	MUTAG	PROTEINS	ENZYMES	IMDB-B	IMDB-M	REDDIT-B	REDDIT-5K	COLLAB	Av.Rank	Av.Gap
Topo-degree	74.20±3.12	55.81±5.67	**87.89±1.29**	65.90±2.48	25.50±3.75	62.88±3.04	41.48±2.73	**84.41±1.73**	**48.34±2.05**	59.52±2.16	7.9	6.2
Topo-HKS	68.52±5.36	60.97±4.22	82.76±3.17	63.34±3.00	25.33±3.40	57.83±2.77	36.32±2.75	78.21±2.83	45.78±1.79	53.50±1.87	10.4	9.5
DOS-hist	68.02±5.39	**68.44±4.69**	73.68±2.88	62.06±1.70	24.04±3.68	64.65±3.53	41.83±2.46	69.84±2.30	38.86±1.90	61.75±1.74	10.6	9.5
DOS-KDE	63.83±5.48	62.74±6.53	78.16±4.09	61.12±2.93	27.58±3.33	65.18±2.87	40.77±3.06	72.85±2.35	39.96±1.50	62.79±2.53	10.3	9.3
Proto-Euc.	67.53±3.69	63.06±6.59	84.61±3.03	64.04±3.74	26.67±2.42	65.45±3.98	40.68±3.57	76.89±3.25	**46.65±2.01**	60.55±2.04	8.0	7.2
Proto-cosine	67.72±5.63	62.31±5.87	84.34±3.17	60.78±4.02	25.88±2.88	66.73±4.24	41.72±2.54	74.00±3.96	42.71±1.74	60.92±3.00	9.4	8.1
GCN	72.41±6.50	58.12±8.48	71.97±4.35	60.87±3.50	24.29±2.03	**68.60±3.31**	**43.82±3.08**	63.70±2.70	41.60±1.86	**65.05±2.59**	8.3	9.8
GAT	53.70±12.89	51.51±21.46	66.84±4.43	58.25±3.46	23.58±2.30	**67.12±4.01**	**43.20±2.95**	60.55±3.06	36.03±4.28	61.52±2.87	13.0	14.6
GSAGE	72.72±2.87	61.13±9.71	70.00±2.68	62.67±3.70	25.33±2.33	66.85±2.57	43.05±3.53	62.58±1.54	41.06±1.86	62.94±2.52	8.9	10.0
GIN	**78.89±4.09**	65.70±5.60	82.11±2.83	63.54±3.39	33.25±3.07	66.65±3.37	**43.82±3.96**	68.64±1.95	39.46±1.23	64.52±2.00	6.2	6.1
GPS	71.05±4.95	65.75±7.49	**88.03±3.86**	**67.42±3.55**	**44.50±3.94**	65.45±5.26	42.35±3.12	74.42±2.97	42.99±1.82	63.93±3.07	**5.2**	**4.2**
GCL	73.77±11.59	**73.87±6.65**	84.47±5.06	60.49±5.36	**36.29±4.62**	65.07±3.50	40.18±3.45	69.41±2.16	38.86±2.48	62.53±3.03	9.1	6.3
SGFormer	55.68±17.52	50.05±9.17	68.16±9.77	61.95±8.83	24.21±2.84	**70.00±2.69**	**43.63±3.85**	60.98±5.00	41.06±2.79	**65.22±2.21**	10.0	12.7
Polynormer	71.30±8.13	56.08±7.58	69.74±2.56	63.00±3.98	27.63±3.05	66.85±4.52	43.03±3.47	69.75±4.27	38.24±2.66	64.33±2.70	8.9	9.8
STAMP-GCN	**75.25±3.59**	61.40±4.76	**86.05±2.89**	**67.35±3.86**	**34.79±3.29**	65.23±3.51	42.67±3.48	**84.05±2.20**	46.05±1.86	63.62±2.53	**5.3**	**4.1**
STAMP-GPS	**75.62±2.50**	**66.56±6.65**	84.47±2.99	**70.09±3.28**	31.54±3.30	65.50±3.83	41.93±3.68	**85.19±1.98**	**46.40±1.25**	**64.99±2.71**	**4.0**	**3.6**

Table 4: Overall performances. The left block (Avg. Rank) reports the average rank of each model across all datasets. The right block (Avg. Gap) reports the average accuracy gap from the best result across all datasets. In each column, the best performance is given in blue, the second best in purple, and the third best in green.

	Average Rank					Average Gap				
Model	**K=1**	**K=5**	**K=10**	**K=25**	**K=50**	**K=1**	**K=5**	**K=10**	**K=25**	**K=50**
Topo-degree	**6.5**	6.5	6.7	7.7	7.9	**4.0**	3.2	**3.0**	4.7	6.2
Topo-HKS	7.9	9.3	10.0	10.0	10.4	**4.1**	4.4	5.8	7.9	9.5
DOS-hist	9.9	8.7	10.4	9.7	10.6	5.7	5.9	7.2	8.2	9.5
DOS-KDE	8.8	8.6	9.1	9.9	10.3	5.9	5.6	6.2	7.8	9.3
Proto-Euc.	9.9	9.5	11.3	10.0	8.0	5.9	5.9	6.8	7.1	7.2
Proto-cosine	**4.4**	**5.7**	**6.2**	8.9	9.4	**2.0**	**3.0**	4.3	6.8	8.1
GCN	7.6	7.7	8.3	8.0	8.4	5.3	5.5	6.9	8.5	9.8
GAT	8.6	11.5	10.6	12.2	13.0	5.6	7.9	8.7	12.6	14.6
GSAGE	8.9	8.3	9.0	8.6	9.0	5.8	6.2	7.0	8.5	10.0
GIN	8.4	7.4	7.9	7.1	6.2	5.2	4.4	4.6	5.4	6.1
GPS	**7.5**	9.4	7.7	**5.5**	**5.2**	5.1	5.8	5.4	**4.2**	**4.2**
GCL	11.2	9.4	9.1	8.6	9.2	6.6	5.9	5.6	5.7	6.3
SGFormer	11.3	9.8	9.9	10.8	10.0	7.4	7.7	9.3	10.3	12.7
Polynormer	8.1	12.3	10.6	9.6	8.9	5.8	8.2	9.2	9.4	9.8
STAMP-GCN	9.2	**5.4**	**4.5**	**5.0**	**5.3**	6.6	**2.6**	**2.0**	**2.7**	**4.1**
STAMP-GPS	7.8	**6.4**	**4.6**	**4.5**	**4.0**	6.2	**2.8**	**1.9**	**2.1**	**3.6**

To ensure a fair and consistent evaluation, we fixed the number of training epochs to 200 across all backbones, including GCN, GAT, GIN, GPS, and the proposed `STAMP` variants. All models were trained with a hidden dimension of 64, a dropout rate of 0.2, a learning rate of 2×10^{-3}, weight decay of 1×10^{-4}, and a batch size of 32. Optimization was performed using the Adam optimizer, with early stopping patience set to 50 epochs, though the final evaluations consistently used the full 200-epoch schedule to eliminate training-length variability.

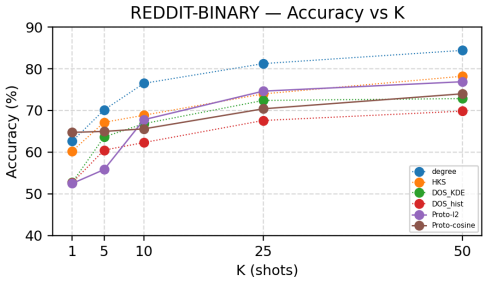
The `STAMP` architecture augments message passing with spectral, topological, and structural embeddings. Topological features were derived from Betti vectorizations with a fixed number of thresholds $T = 10$, while structural embeddings captured node degree distributions. Spectral embeddings were parameterized using 128 bins for DOS histograms and $m = 128$ grid points for DOS-KDE, with a Gaussian kernel bandwidth of 0.01. All spectral, topological, and structural descriptors were standardized to zero mean and unit variance prior to integration to ensure balanced scaling and stable training dynamics.

We evaluate two variants of `STAMP`: `STAMP-GPS`, which leverages a GPS backbone for attention-enhanced message passing, and `STAMP-GCN`, which utilizes a simpler GCN backbone for a lighter-weight alternative. This design enables a fair analysis of the benefits of spectral-topological augmentation across different levels of architectural expressiveness. Our implementation is available at `https://github.com/1999-karthik/STAMP.git`

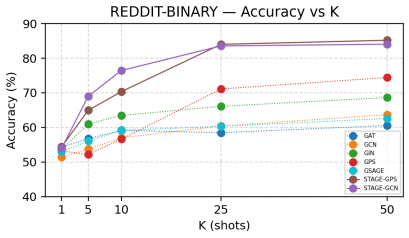
4.1 Results

In Tables 3 and 4, we present both per-dataset accuracies and cross-dataset summaries for all methods across five label budgets $K \in \{1, 5, 10, 25, 50\}$. Table 3 reports accuracy on each TU dataset, while Table 4 aggregates performance using two complementary metrics with lower being better: *Average Rank* (the mean rank of a method across datasets within each K column) and *Average Gap* (the mean absolute difference in percentage points to the column-best accuracy). We compare four categories: direct nonparametric descriptors (Topo-Degree, Topo-HKS, DOS-hist, DOS-KDE), our prototype embeddings (Euclidean and cosine) that combine these descriptors, standard parametric baselines (GCN, GAT, GSAGE, GIN, GPS), and our hybrids `STAMP-GCN` and `STAMP-GPS`.

Low-label regimes. Across ten TU benchmarks and label budgets $K \in \{1, 5, 10, 25, 50\}$, our *prototype embeddings* are strongest when labels are extremely scarce. The cosine variant, which fuses all direct descriptors (Topo-Degree, Topo-HKS, DOS-hist, DOS-KDE) into a similarity-to-prototype space, attains the best average ranks at $K{=}1$ and $K{=}5$ and the smallest or near-smallest average gaps in Table 4. Topo-Degree alone is often second or third. Standard deep baselines trail by a wide margin in this regime, confirming that aggregating multiple stable structural views benefits generalization when supervision is minimal.



(a) Direct Embeddings



(b) Deep Models

Figure 3: Reddit-B: Performance Evolution. Performance evolution for direct embeddings (left) and deep models (right) in low-data settings on the Reddit-B dataset. The corresponding figures for other datasets are given in Appendix B.

Growing label budgets. As K increases, the `STAMP` hybrids dominate. Beginning at $K{=}10$, `STAMP-GPS` achieves the best overall rank and the lowest average gap and remains on top at $K{=}25$ and $K{=}50$. `STAMP-GCN` shows the same pattern, slightly behind `STAMP-GPS` yet ahead of all non-hybrid methods. These gains indicate that conditioning message passing on global spectral and topological descriptors yields a clear sample-efficiency benefit once a modest number of labels are available.

Takeaway. Use nonparametric structure when labels are tiny, then switch to the hybrid. Our prototype embeddings, which combine all direct descriptors into a unified prototype space, are the most effective choice for $K \leq 5$. For $K \geq 10$, `STAMP` consistently delivers the best overall accuracy and rank. These trends are clearly illustrated in Figures 3, 4 and 5, where the transition from prototype embeddings to `STAMP` as K grows is visually evident.

Limitations and future work. Our descriptors assume access to reliable structural cues and require a modest amount of spectral computation; very large graphs may benefit from additional accelerations. Extending the approach to dynamic, attributed, and heterogeneous graphs, and combining `STAMP` with active learning or semi-supervised objectives, are promising directions. Another avenue is to study sample complexity and representation alignment more formally, and to explore synergy with self-supervised pretraining. We hope this structure-aware perspective encourages the community to treat nonparametric graph descriptors as first-class citizens in low-shot learning.

5 Conclusion

We revisited graph classification under small label budgets and showed that structure first methods are a powerful and complementary alternative to purely deep approaches. We evaluated two label-free families of descriptors, topological vectors from persistent homology and spectral vectors from the Laplacian density of states, and introduced prototype embeddings that combine these views in a unified similarity space. We further proposed `STAMP`, a compact controller that conditions GNN and Graph Transformer backbones on these descriptors through simple concatenation and layerwise modulation. Across ten TU benchmarks and $K \in \{1, 5, 10, 25, 50\}$, prototype embeddings dominate in the extreme low label regime, while STAMP achieves the best overall rank and the lowest accuracy gaps once $K \geq 10$. Together with stability bounds and ablations, these results establish that explicit structural priors can materially improve label efficiency without pretraining.

Practically, our findings suggest a simple recipe for small data settings. Start with a nonparametric structure when labels are tiny, using prototype embeddings that fuse topological and spectral signals. As labels grow, switch to the STAMP hybrid, which retains the robustness of explicit descriptors while leveraging task-specific adaptation through message passing. The method is lightweight, easy to integrate into standard backbones, and yields consistent gains across heterogeneous datasets.

Acknowledgements. This research was partially supported by the National Science Foundation under grants DMS-2220613 and DMS-2229417. The authors thank Prof. Lakshman Tamil for insightful discussions. The authors acknowledge the Texas Advanced Computing Center (TACC) at UT Austin for providing computational resources that have contributed to this paper.

References

[1] Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. *arXiv preprint arXiv:2007.08663*, 2020. URL `https://chrsmrrs.github.io/datasets`. 1, 6

[2] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1

[3] Robert L Peach, Alexis Arnaudon, Julia A Schmidt, Henry A Palasciano, Nathan R Bernier, Kim E Jelfs, Sophia N Yaliraki, and Mauricio Barahona. Hcga: Highly comparative graph analysis for network phenotyping. *Patterns*, 2(4), 2021. 1

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

[5] Anton Tsitsulin, Davide Mottin, Panagiotis Karras, Alex Bronstein, and Emmanuel Müller. Netlsd: Hearing the shape of a graph. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, pages 2347–2356, 2018. 1, 2, 3

[6] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Stability of persistence diagrams. *Discrete & Computational Geometry*, 37(1):103–120, 2007. 2, 3

[7] Peter Bubenik. Statistical topological data analysis using persistence landscapes. *Journal of Machine Learning Research*, 16:77–102, 2015.

[8] Mathieu Carrière, Frédéric Chazal, Yuichi Ike, Théo Lacombe, Martin Royer, and Yuhei Umeda. Perslay: A neural network layer for persistence diagrams and new graph topological signatures. In *AISTATS*, pages 2786–2796, 2020. 2

[9] Kun Dong, Austin R. Benson, and David Bindel. Network density of states. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, pages 1152–1161, 2019. 2, 3

[10] Lin Lin, Yousef Saad, and Chao Yang. Approximating spectral densities of large matrices. *SIAM Review*, 58(1):34–65, 2016. 2, 3

[11] Shunyu Jiang, Fuli Feng, Weijian Chen, Xiang Li, and Xiangnan He. Structure-enhanced meta-learning for few-shot graph classification. *AI Open*, 2:160–167, 2021. 2

[12] Song Wang, Yushun Dong, Xiao Huang, Chen Chen, and Jundong Li. Faith: Few-shot graph classification with hierarchical task graphs. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2284–2290, 2022. 2

[13] Donato Crisostomi et al. Metric based few-shot graph classification. In *Proceedings of the First Learning on Graphs Conference (LoG)*, volume 198, pages 33:1–33:22. PMLR, 2022. 2

[14] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2

[15] Ladislav Rampášek, Mikhail Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[16] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay S. Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. In *International Conference on Learning Representations (ICLR)*, 2020. 2

[17] Federico Errica, Marco Podda, Davide Bacciu, and Alessio Micheli. A fair comparison of graph neural networks for graph classification. In *ICLR*, 2020. 2

[18] Christoph Hofer, Florian Graf, Bastian Rieck, Marc Niethammer, and Roland Kwitt. Graph filtration learning. In *International Conference on Machine Learning*, pages 4314–4323. PMLR, 2020. 2

[19] Phu Pham, Quang-Thinh Bui, Ngoc Thanh Nguyen, Robert Kozma, Philip S Yu, and Bay Vo. Topological data analysis in graph neural networks: Surveys and perspectives. *IEEE Transactions on Neural Networks and Learning Systems*, 2025. 2

[20] Bastian Rieck, Christian Bock, and Karsten Borgwardt. A persistent weisfeiler-lehman procedure for graph classification. In *International Conference on Machine Learning*, pages 5448–5458. PMLR, 2019. 2

[21] Qi Zhao, Ze Ye, Chao Chen, and Yusu Wang. Persistence enhanced graph neural network. In *International Conference on Artificial Intelligence and Statistics*, pages 2896–2906. PMLR, 2020. 2

[22] Chaolong Ying, Xinjian Zhao, and Tianshu Yu. Boosting graph pooling with persistent homology. *Advances in Neural Information Processing Systems*, 37:19087–19113, 2024. 2

[23] Andac Demir, Baris Coskunuzer, Yulia Gel, Ignacio Segovia-Dominguez, Yuzhou Chen, and Bulent Kiziltan. Todd: Topological compound fingerprinting in computer-aided drug discovery. *NeurIPS*, 35:27978–27993, 2022. 2

[24] Zuoyu Yan, Tengfei Ma, Liangcai Gao, Zhi Tang, Yusu Wang, and Chao Chen. Neural approximation of graph topological features. *Advances in neural information processing systems*, 35:33357–33370, 2022. 2

[25] Minghua Wang, Yan Hu, Ziyun Huang, Di Wang, and Jinhui Xu. Persistent local homology in graph learning. *Transactions on Machine Learning Research*, 2024. 2

[26] Andries E. Brouwer and Willem H. Haemers. *Spectra of Graphs*. Universitext. Springer, New York, 2012. 3

[27] Fan R.K. Chung. *Spectral Graph Theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, 1997. 3, 14

[28] Shashanka Ubaru, Jie Chen, and Yousef Saad. Fast estimation of tr(f(a)) via stochastic lanczos quadrature. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1075–1099, 2017. 3

[29] David K. Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011. 3

[30] David I. Shuman et al. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013.

[31] Feng Gao, Guy Wolf, and Matthew Hirn. Geometric scattering for graph data analysis. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2122–2131. PMLR, 2019.

[32] Feng Xia, Ke Sun, Shuo Yu, Abdul Aziz, Liangtian Wan, Shirui Pan, and Huan Liu. Graph learning: A survey. *IEEE Transactions on Artificial Intelligence*, 2(2):109–127, 2021. 3

[33] Tamal Krishna Dey and Yusu Wang. *Computational Topology for Data Analysis*. Cambridge University Press, 2022. 3, 13

[34] Baris Coskunuzer and Cüneyt Gürcan Akçora. Topological methods in machine learning: A tutorial for practitioners. *arXiv preprint arXiv:2409.02901*, 2024. 3, 13

[35] Dashti Ali et al. A survey of vectorization methods in topological data analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 3

[36] Daniel A Spielman. Spectral graph theory and its applications. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 29–38. IEEE, 2007. 3

[37] Nils Kriege et al. Subgraph matching kernels for attributed graphs. In *ICML*, pages 291–298, 2012. 6

[38] Pinar Yanardag and SVN Vishwanathan. Deep graph kernels. In *KDD*, pages 1365–1374, 2015. 6

[39] Primoz Skraba and Katharine Turner. Wasserstein stability for persistence diagrams. *arXiv:2006.16824*, 2020. 13

[40] Paweł Dłotko and Davide Gurnari. Euler characteristic curves and profiles: a stable shape invariant for big data problems. *GigaScience*, 12:giad094, 2023. 13

[41] Megan Johnson and Jae-Hun Jung. Instability of the Betti sequence for persistent homology and a stabilized version of the Betti sequence. *Journal of the Korean Society for Industrial and Applied Mathematics*, 25(4):296–311, 2021. 13

[42] Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17:395–416, 2007. 14

Appendix

A Proofs of Stability Theorems

In this part, we prove the stability results for our descriptors.

A.1 Stability of Topological Descriptors.

To establish the stability of Betti vectors, we begin by recalling two fundamental results from applied topology. The first lemma addresses the stability of persistence diagrams under sublevel filtrations, while the second focuses on the stability of Betti vectorizations with respect to the L^1-norm. In the following, $\|\cdot\|_p$ represents L^p-norm, and $\mathcal{W}_p(\cdot,\cdot)$ represents the Wasserstein distance [33, 34].

Lemma A.1. *[39] Let $\mathcal{X}$ be a compact metric space, and $f, g : \mathcal{X} \to \mathbb{R}$ be two filtration functions. Then, for any $p \geq 1$, we have* $\mathcal{W}_p(\mathrm{PD_k}(\mathcal{X}, \mathrm{f}), \mathrm{PD_k}(\mathcal{X}, \mathrm{g})) \leq \|\mathrm{f} - \mathrm{g}\|_\mathrm{p}$.

The next lemma is on the stability of Betti curves by [40].

Lemma A.2. *[40] Let $\beta_k(\mathcal{X})$ is the k^{th} Betti function obtained from the persistence module $\mathrm{PM_k}(\mathcal{X})$. Then,* $\|\vec{\beta}_k(\mathcal{X}) - \vec{\beta}_k(\mathcal{Y})\|_1 \leq 2\mathcal{W}_1(\mathrm{PD_k}(\mathcal{X}), \mathrm{PD_k}(\mathcal{Y}))$ *where $\vec{\beta}_k(\mathcal{X})$ represents the Betti curves corresponding to $\mathrm{PD_k}(\mathcal{X})$*

Now, we are ready to prove our stability result for Betti vectors. In the following, let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph and let $f, g : \mathcal{V} \to \mathbb{R}$ be two filtration functions. Fix the number of thresholds N and width parameter m as introduced in Section 2.2. Define the thresholds as N-quantiles of the sets $f(\mathcal{V})$ and $g(\mathcal{V})$, and define the subgraphs $\{\mathcal{G}_i\}_{i=1}^N$ from filtration process accordingly. Let $\widehat{\mathcal{G}}_i$ be the clique complex of $\mathcal{G}_i$. Let $\widehat{\beta}_k(\mathcal{G}, f), \widehat{\beta}_k(\mathcal{G}, g)$ be the N-dimensional Betti vectors for filtration functions f, g, respectively.

Theorem A.3. *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph and let $f, g : \mathcal{V} \to \mathbb{R}$ be two filtration functions. Then, for $k \geq 0$, we have* $\|\widehat{\beta}_k(\mathcal{G}, f) - \widehat{\beta}_k(\mathcal{G}, g)\|_1 \leq C_k \cdot \|f - g\|_1$.

Proof. The proof follows from the results above. Let $\widehat{\mathcal{G}}$ be the clique complex of $\mathcal{G}$. Then, by using the filtration functions $f, g : \mathcal{V} \to \mathbb{R}$, we can induce a function on the clique complex $\widehat{f}, \widehat{g} : \widehat{\mathcal{G}} \to \mathbb{R}$ such that for any k-simplex $\sigma = [v_{i_0}, v_{i_1}, \ldots, v_{i_k}]$, define $\widehat{f}(\sigma) = \max_{v_i \in \sigma}\{f(v_i)\}$ ([39]-Section 2). Then, by Lemma A.1, we have

$$\mathcal{W}_1(\mathrm{PD_k}(\widehat{\mathcal{G}}, \mathrm{f}), \mathrm{PD_k}(\widehat{\mathcal{G}}, \mathrm{g})) \leq \|\mathrm{f} - \mathrm{g}\|_1 \tag{1}$$

Next by using simplicial complex $\widehat{\mathcal{G}}$ as the compact space and $\widehat{f}, \widehat{g}$ as simplicial maps, in Lemma A.2, we obtain persistence modules $\mathrm{PM_k}(\mathcal{G}, \mathrm{f})$ and $\mathrm{PM}(\mathcal{G}, \mathrm{g})$, which gives

$$\|\vec{\beta}_k(\widehat{\mathcal{G}}, f) - \vec{\beta}_k(\widehat{\mathcal{G}}, g)\|_1 \leq 2\mathcal{W}_1(\mathrm{PD_k}(\widehat{\mathcal{G}}, \mathrm{f}), \mathrm{PD_k}(\widehat{\mathcal{G}}, \mathrm{g})) \tag{2}$$

Hence, by combining Equations (1) and (2), the proof follows. □

We note that L^1-stability result above does not extend to L^∞-norm [41].

Stability of Spectral Descriptors. Next, we prove the stability of the density of states (DoS) vectors under edge modifications.

Theorem 3.2. *Let $\mathcal{G}$ and $\mathcal{G}'$ be two graphs differing by at most k edge modifications (insertions or deletions). Then, the Wasserstein distance between the spectral descriptors ψ and ψ' is bounded by Ck/n, where $n = |\mathcal{V}|$ and C is a constant depending on the eigenvalue distribution.*

Proof. Let $\mathcal{G}$ and $\mathcal{G}'$ be two graphs differing by at most k edge insertions or deletions. Let ψ, ψ' be the corresponding spectral descriptors.

Next, we recall the following results from spectral graph theory.

- Adding or deleting an edge modifies at most two entries in the normalized Laplacian matrix.
- The matrix perturbation theory for eigenvalues (see, e.g., [42]) implies that if L and L' differ by a matrix of spectral norm at most ϵ, then the eigenvalues of L and L' differ by at most ϵ.
- Since each edge modification changes at most $O(1/n)$ in spectral norm [27], k edge changes induce a perturbation of size at most $O(k/n)$.
- Therefore, the Wasserstein (matching) distance between the DOS histograms is bounded as $\mathcal{W}_1(\psi, \psi') = O\left(\frac{k}{n}\right)$, where $\mathcal{W}_1$ denotes the 1-Wasserstein distance.

This implies $\mathcal{W}_1(\psi, \psi') \leq Ck/n$ for some $C > 0$. The proof follows. □

B Additional Performance Evolution Figures

In this section, we present the full set of performance–evolution figures for all benchmark datasets. For each dataset, results are shown separately for direct embeddings (left) and deep models (right), with accuracy plotted against the number of labels per class $K \in \{1, 5, 10, 25, 50\}$. These figures complement the REDDIT-B example in the main text and provide a detailed view of how different model families behave under varying label budgets. Together, they highlight the regimes where nonparametric descriptors, prototype embeddings, and STAMP hybrids are most effective.

Figure 4: Accuracy vs. # training samples. Performance evolution for direct embeddings (left) and deep models (right) in low-data settings on benchmark datasets.

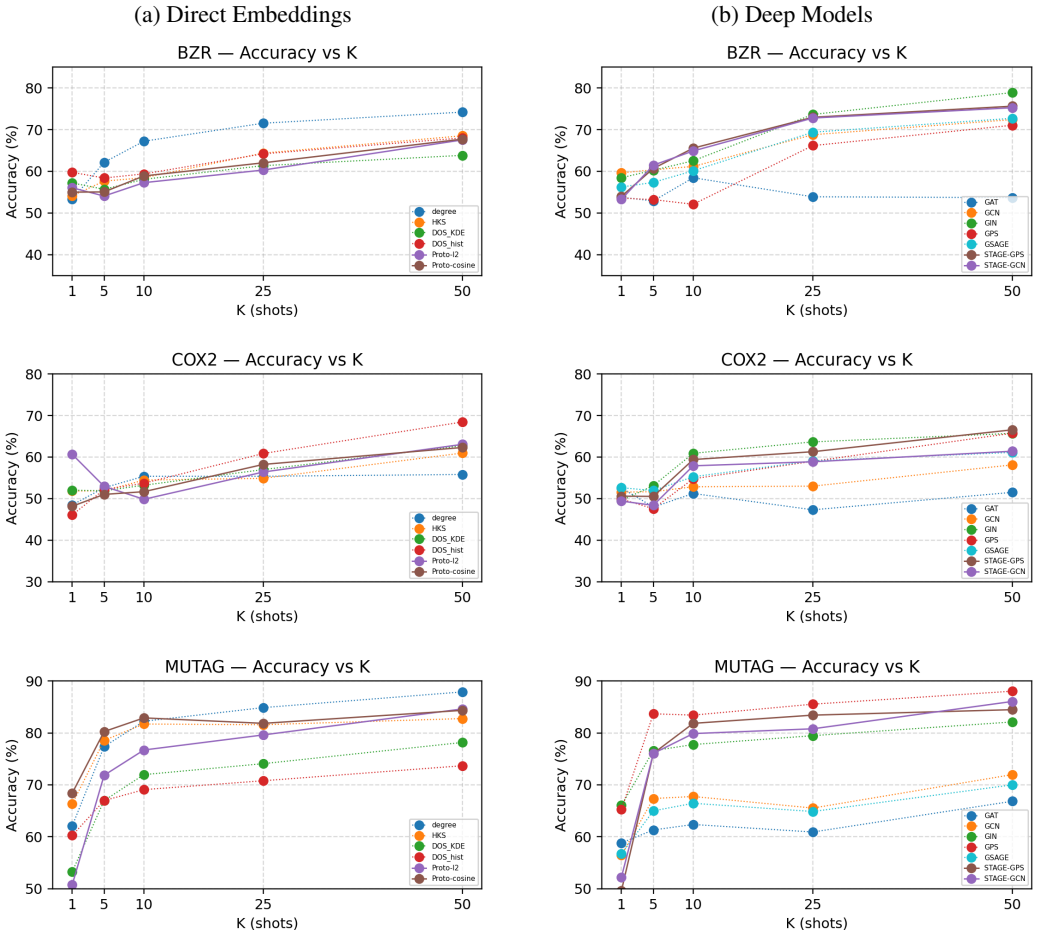
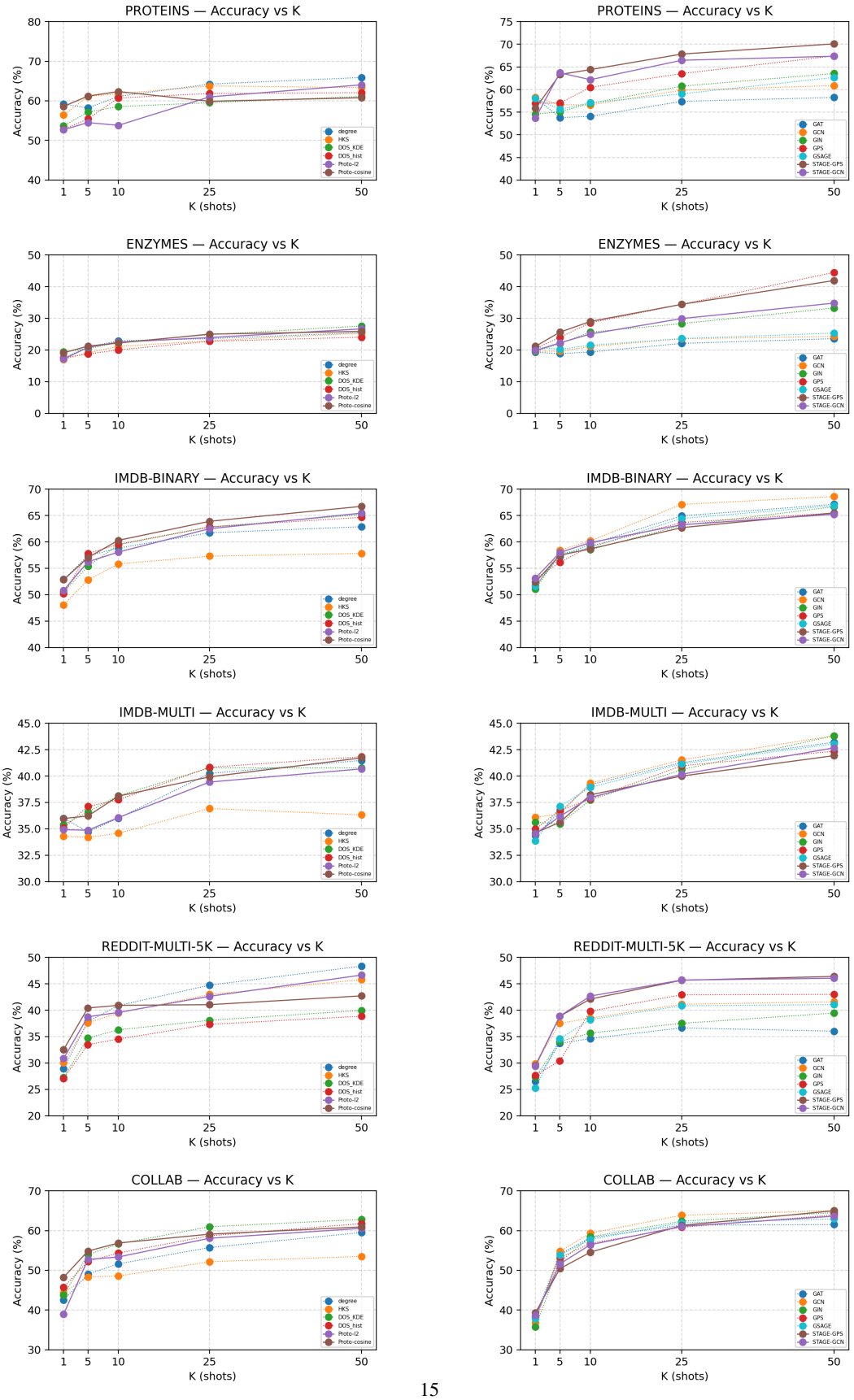


Figure 5: Accuracy vs. # training samples. Performance evolution for direct embeddings (left) and deep models (right) in low-data settings on benchmark datasets.



C STAMP Training and Forward Pass

In this part, we provide details on how STAMP is implemented and optimized. We break the method into three parts and provide self-contained pseudocode for each. First, we *precompute and standardize* the global descriptors for every graph: Betti vectors x_G and DOS vectors y_G are computed once, and z-score parameters are fitted on the training split only, then reused for validation and test. Second, a small *controller* encodes the standardized descriptors (z_x, z_y) into a global context g that is shared across all layers. Third, each backbone layer is *modulated* by g through biasing, gating, and affine normalization, after which pooled node embeddings are concatenated with (z_x, z_y) for classification. Reusable routines for these steps are given in Algorithm 1; the full mini-batch training loop appears in Algorithm 2; and the inference-time forward pass is summarized in Algorithm 3.

The algorithms expose common choices as arguments: the backbone ($\mathsf{Backbone} \in \{\mathrm{GCN}, \mathrm{GPS}\}$), normalization ($\mathsf{Norm} \in \{\mathrm{GN}, \mathrm{LN}, \mathrm{BN}\}$), and pooling ($\mathsf{Pool} \in \{$mean,sum,max,JK$\}$). Two scalars control modulation strength: s scales the affine parameters (γ, β) and λ_t gates residual contributions; we treat λ_t as a schedule during training for flexibility but fix it to the constant value 0.25 in all experiments for stability and reproducibility. Unless otherwise noted, positional encodings are zero-filled if absent, and all standardization statistics are computed strictly on the training set.

Algorithm 1 Reusable Procedures for STAMP

function PRECOMPUTEDESCRIPTORS($\mathcal{S}, N_{\text{th}}, M$)
  for all $G \in \mathcal{S}$ **do**
    $x_G \leftarrow \text{BettiVector}(G, N_{\text{th}})$ ▷ $x_G \in \mathbb{R}^{2N_{\text{th}}}$
    $y_G \leftarrow \text{DOSVector}(G, M)$ ▷ $y_G \in \mathbb{R}^M$
  end for
end function
function FITSTANDARDIZERS($\mathcal{S}_{\text{train}}$)
  $(\mu_x, \sigma_x) \leftarrow \text{FitZScore}(\{x_G\}_{G \in \mathcal{S}_{\text{train}}})$
  $(\mu_y, \sigma_y) \leftarrow \text{FitZScore}(\{y_G\}_{G \in \mathcal{S}_{\text{train}}})$
  return $(\mu_x, \sigma_x, \mu_y, \sigma_y)$
end function
function STANDARDIZEGLOBALS($x_G, y_G, \mu_x, \sigma_x, \mu_y, \sigma_y$)
  $z_x \leftarrow (x_G - \mu_x)/\sigma_x; \quad z_y \leftarrow (y_G - \mu_y)/\sigma_y$
  return (z_x, z_y)
end function
function CONTROLLER(z_x, z_y)
  return $\mathrm{ReLU}(\mathrm{LayerNorm}(W_g[z_x \,\|\, z_y]))$
end function
function LAYERSTEP($h, g, \ell, E, b, \mathsf{Backbone}, \mathsf{Norm}, s, \lambda_t$)
  $m \leftarrow \mathsf{BackboneLayer}_\ell(h, E, b)$
  $u \leftarrow m \odot \big(1 + \tanh(W^\ell_{\text{bias}} g)[b]\big)$
  $u \leftarrow \big(1 + \lambda_t \cdot \sigma(W^\ell_{\text{gate}} g)[b]\big) \odot u$
  $\hat{u} \leftarrow \mathsf{Norm}(u)$
  $\gamma \leftarrow 1 + s \cdot \tanh(W^\ell_\gamma g); \quad \beta \leftarrow s \cdot \tanh(W^\ell_\beta g)$
  return $u + (\gamma[b] - 1) \odot \hat{u} + \beta[b]$
end function
function POOLANDCLASSIFY(h, b, z_x, z_y)
  $z \leftarrow \mathsf{Pool}(h, b)$
  return $\mathrm{MLP}(z \,\|\, z_x \,\|\, z_y)$
end function

Algorithm 2 Training STAMP

Require: $\mathcal{D}$; N_{th}; M; H; L; s; schedule λ_t (default constant 0.25);
Require: Norm $\in$ $\{\text{GraphNorm}, \text{LayerNorm}, \text{BatchNorm}\}$; Pool $\in$ {mean,sum,max,JK}; Backbone $\in$ {GCN,GPS}

Split $\mathcal{D}$ into $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{val}}, \mathcal{D}_{\text{test}}$
PRECOMPUTEDESCRIPTORS($\mathcal{D}_{\text{train}}, N_{\text{th}}, M$); PRECOMPUTEDESCRIPTORS($\mathcal{D}_{\text{val}}, N_{\text{th}}, M$); PRECOMPUTEDESCRIPTORS($\mathcal{D}_{\text{test}}, N_{\text{th}}, M$)
$(\mu_x, \sigma_x, \mu_y, \sigma_y) \leftarrow$FITSTANDARDIZERS($\mathcal{D}_{\text{train}}$)
Initialize controller, modulation heads, backbone, classifier
for epoch $= 1, \ldots$ **do**
  for mini-batch $\mathcal{B} \subset \mathcal{D}_{\text{train}}$ **do**
    build (X, E, b); gather $\{x_G, y_G\}_{G \in \mathcal{B}}$
    $(z_x, z_y) \leftarrow$STANDARDIZEGLOBALS($x_G, y_G, \mu_x, \sigma_x, \mu_y, \sigma_y$)
    $g \leftarrow$CONTROLLER(z_x, z_y)
    ensure PE (zeros if absent); $h \leftarrow \text{Linear}(X \,\|\, \text{PE})$
    for $\ell = 1$ to L **do**
      $h \leftarrow$LAYERSTEP($h, g, \ell, E, b,$ Backbone, Norm, s, λ_t)
      if $\ell < L$ **then**
        $h \leftarrow \text{ReLU}(h)$; $h \leftarrow \text{Dropout}(h)$
      end if
    end for
    $o \leftarrow$POOLANDCLASSIFY(h, b, z_x, z_y); $\mathcal{L} \leftarrow \text{loss}(o, y)$
    Backprop & optimizer step
  end for
end for
return best checkpoint by validation performance
Note: in all experiments we use constant $\lambda_t \equiv 0.25$.

Algorithm 3 Forward Pass of STAMP

Require: $(X, E, b, \{x_G, y_G\})$; (μ_x, σ_x), (μ_y, σ_y); L, s, λ_t

$(z_x, z_y) \leftarrow$STANDARDIZEGLOBALS($x_G, y_G, \mu_x, \sigma_x, \mu_y, \sigma_y$)
$g \leftarrow$CONTROLLER(z_x, z_y); ensure PE; $h \leftarrow \text{Linear}(X \,\|\, \text{PE})$
for $\ell = 1$ to L **do**
  $h \leftarrow$LAYERSTEP($h, g, \ell, E, b,$ Backbone, Norm, s, λ_t)
  if $\ell < L$ **then**
    $h \leftarrow \text{ReLU}(h)$; $h \leftarrow \text{Dropout}(h)$
  end if
end for
return POOLANDCLASSIFY(h, b, z_x, z_y)