A SPECTRAL FRAMEWORK FOR EVALUATING GEODESIC DISTANCES BETWEEN GRAPHS

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper presents a spectral framework for quantifying the differentiation between graph data samples by introducing a novel metric named Graph Geodesic Distance (GGD). For two different graphs with the same number of nodes, our framework leverages a spectral graph matching procedure to find node correspondence so that the geodesic distance between them can be subsequently computed by solving a generalized eigenvalue problem associated with their Laplacian matrices. For graphs of different sizes, a resistance-based spectral graph coarsening scheme is introduced to reduce the size of the larger graph while preserving the original spectral properties. We show that the proposed GGD metric can effectively quantify dissimilarities between two graphs by encapsulating their differences in key structural (spectral) properties, such as effective resistances between nodes, cuts, and the mixing time of random walks. Through extensive experiments comparing with state-of-the-art metrics, such as the latest Tree-Mover's Distance (TMD), the proposed GGD metric demonstrates significantly improved performance for graph classification, particularly when only partial node features are available. Furthermore, we extend the application of GGD beyond graph classification to stability analysis of GNNs and the quantification of distances between datasets, highlighting its versatility in broader machine learning contexts.

1 INTRODUCTION

In the era of big data, comparison and distinction between data points are important tasks. A graph is a specific type of data structure that represents the connections between a group of nodes. Comparing two graphs often involves using a pairwise distance measure, where a small distance indicates a high structural similarity and vice versa. To understand the generalization between distribution shifts, it is important to use an appropriate measure of divergence between data distributions, both theoretically and experimentally (Chuang et al., 2020). Determining suitable distance metrics for non-Euclidean data, such as graphs with or without node attributes, remains challenging. These metrics are fundamental to many graph learning methods, such as graph neural networks (GNNs), but are not as readily available as those for Euclidean space. The need to develop new analytical techniques that allow the visualization, comparison, and understanding of different graphs has led to a rich field of research (Haslbeck & Waldorp, 2018). This study dives into the exploration of a novel framework for computing geometric distances between graphs, which can be immediately leveraged for many graph-based machine learning (ML) tasks.

Many distance metrics for comparing graphs have previously been proposed (Borgwardt et al., 2020). Some of them are based only on graph local structures (Tam & Dunson, 2022; Haussler et al., 1999; Xu et al., 2013; Zhu et al., 2020; Fernández & Valiente, 2001; Bunke & Shearer, 1998), whereas others exploit both graph structural properties and node attributes (Shervashidze et al., 2011; Morris et al., 2019). For example, the Graph Edit Distance (GED) has been proposed to measure the distance between graphs considering the number of changes needed to match one graph to another (Sanfeliu & Fu, 1983; Gao et al., 2010; Li et al., 2017); Distance metrics based on the graph kernel have also been investigated (Shervashidze et al., 2011; Vishwanathan et al., 2010), such as the Wasserstein Weisfeiler-Leman metric (WWL) (Morris et al., 2019) and the Gromov–Wasserstein metric (Mémoli, 2011), which allow computing graph distances based on low-dimensional graph representations or optimal transport (OT) (Titouan et al., 2019; Chapel et al., 2020), leading to the development of the state-of-the-art graph distance metric called TMD (Chuang & Jegelka, 2022).

However, existing graph distance metrics have notable limitations. For example, the GED metric can capture local node or edge changes but struggles with global perturbations (Sanfeliu & Fu, 1983; Gao et al., 2010; Li et al., 2017); the WWL and TMD metrics heavily rely on node features (attributes) for calculating the distance between graphs, leading to degraded performance when only partial node features are available (Rossi et al., 2022; Chen et al., 2022).

To address these limitations, we propose the **Graph Geodesic Distance** (GGD) metric, a novel framework that leverages spectral graph theory, structure-preserving coarsening, and Riemannian geometry to compute meaningful distances between graphs. Unlike prior SPD-based works (Lim et al., 2019), our method operates directly on graph inputs, using spectral graph matching to establish node correspondence before embedding graphs into a Riemannian manifold of modified Laplacian matrices. This allows GGD to capture key structural (spectral) dissimilarities between graphs, such as mismatches in Laplacian eigenvalues/eigenvectors, cuts, effective-resistance distances, etc.

One distinct advantage of the proposed GGD metric is its capability to compute distances between graphs based on their spectral (structural) properties, while including node features can further improve its accuracy. This makes GGD suitable for analyzing real-world graphs with partial or no node features. Moreover, the proposed framework for computing GGD metric is more computationally efficient than existing OT-based metrics, such as the TMD metric.

Our empirical results show that GGD can effectively measure the dissimilarities between graphs: (1) support vector classifiers (SVC) using GGDs perform competitively with state-of-the-art GNN models and graph kernels on graph classification benchmarks; (2) we demonstrate that the GGD metric allows us to quantify the stability of GNN models for graph classification tasks by checking whether two graphs with a small GGD will lead to a significant dissimilarity in the GNN output embeddings; (3) quantify distance between datasets to evaluate the transferability of domain knowledge. We also show that the GGD metric has a better correlation with established GNN outputs compared to the state-of-the-art TMD metric (Chuang & Jegelka, 2022) when only partial node features are available: up to a 10% accuracy gain and a $9\times$ runtime speedup have been achieved in various graph classification tasks. The key contributions of this work can be summarized as:

- We propose Graph Geodesic Distance (GGD), a novel metric combining spectral graph theory and Riemannian geometry.
- We leverage a spectral graph matching method to establish node correspondences to ensure well-defined distance computation in the Riemannian manifold.
- We use a resistance-based spectral coarsening method to compare graphs of different sizes.
- We validate GGD through experiments, demonstrating improved classification accuracy with partial/no features and applications in GNN stability and dataset distance analysis.

2 Existing Graph Distance Metrics

Graph Edit Distance (GED) For non-attributed graph data, a common and simple distance metric is GED. (Sanfeliu & Fu, 1983; Gao et al., 2010). Given a set of graph edit operations, also known as elementary graph operations, the GED between two graphs G_1 and G_2, can be defined as:

$$\mathrm{GED}(G_1, G_2) = \min_{(e_1,\ldots,e_k)\in\mathcal{P}(G_1,G_2)} \sum_{i=1}^{k} c(e_i), \tag{1}$$

where $\mathcal{P}(G_1, G_2)$ denotes the set of edit operations transforming G_1 into a graph isomorphism of G_2, $c(e_i)$ is the cost of edit operation e_i. The set of elementary graph edit operators typically includes node insertion, node deletion, node substitution, edge insertion, edge deletion, and edge substitution.

Tree Mover's Distance (TMD) TMD is a pseudometric for measuring distances between simple graphs, extending the concept of WWL to multisets of tree structures (Chuang & Jegelka, 2022). Graphs are represented by depth-L node computation trees; unequal multiset sizes are equalized via augmentation function σ. The pairwise tree distance TD_w is defined recursively by combining the root-feature discrepancy with a depth-weighted OT over subtrees. The TMD between G_1, G_2 is:

$$\mathrm{TMD}_w^L(G_1, G_2) = \mathrm{OT}_{\mathrm{TD}_w}(\sigma(\mathbf{T}_{G_1}^L, \mathbf{T}_{G_2}^L)). \tag{2}$$

Further details on TMD is described in Appendix A.2.

3 GGD: Geodesic Distance Between Graphs

Modified Laplacian matrices on the Riemannian manifold One way to represent a simple connected graph is through its Laplacian matrix, which is a Symmetric Positive Semidefinite (SPSD) matrix. Graph representation using adjacency and Laplacian matrices is briefly discussed in Appendix A.3. Adding a small positive value to each diagonal element allows us to transform the original Laplacian matrix into a Symmetric Positive Definite (SPD) matrix, which is referred to as the **Modified Laplacian Matrix** in this work. In Appendix A.10, we describe the effect of this small value on the GGD calculation. We can then consider the cone of such modified Laplacian matrices as a natural Riemannian manifold (Lim et al., 2019), where each modified Laplacian, having the same dimensions (same number of rows/columns), can be regarded as a data point on this Riemannian manifold (Vemulapalli & Jacobs, 2015; Pennec et al., 2006). Details about the Riemannian manifold are provided in Appendix A.4. The geodesic distance is defined as the shortest path on the Riemannian manifold, providing a more appropriate comparison than Euclidean space (Lim et al., 2019; Crane et al., 2020; Huang et al., 2015). We will later demonstrate (Section 4.3) that such a geodesic distance metric can effectively capture structural (spectral) mismatches between graphs.

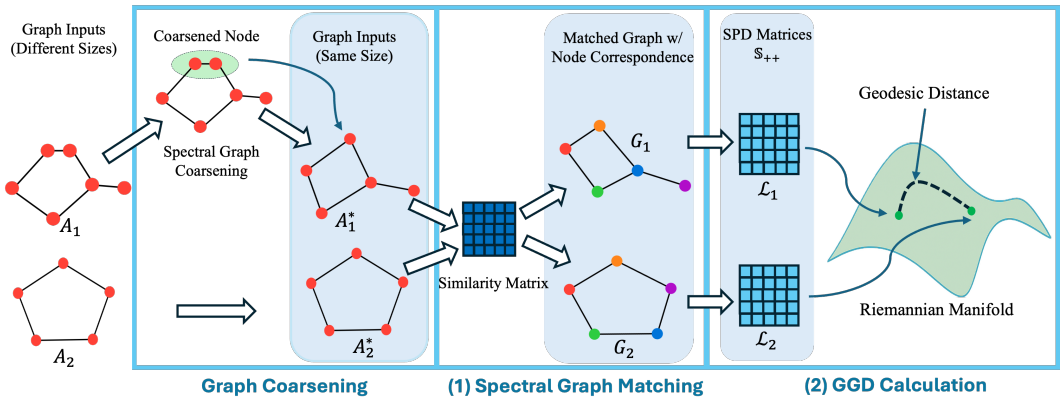

Figure 1: A high-level illustration of the GGD pipeline, including spectral graph coarsening, (Phase 1) spectral graph matching, and (Phase 2) geodesic distance calculation for the Riemannian manifold of modified Laplacian matrices.

A two-phase spectral framework for computing GGDs Before computing GGDs, it is necessary to establish the node-to-node correspondence between two graphs. This can be achieved by leveraging existing graph-matching techniques (Livi & Rizzi, 2013; Emmert-Streib et al., 2016; Caetano et al., 2009). The proposed GGD metric can be computed in the following two phases. **Phase 1** consists of a spectral graph matching step, using combinatorial optimization with the eigenvalues/eigenvectors of the graph adjacency matrices to identify the approximate node-to-node correspondence. **Phase 2** computes the GGD between the modified Laplacian matrices of the matched graphs by exploiting generalized eigenvalues. A high-level overview of the pipeline is illustrated in Figure 1, and a detailed algorithmic flow is provided in Appendix A.1 to ensure a clear understanding of the process. The proposed GGD metric differs from previous OT-based graph distance metrics in its ability to accurately represent structural discrepancies between graphs, enabling us to uncover the topological variations between them more effectively.

A motivating example Let's consider a simple graph G_1, characterized by an almost ring-like topology, as shown in Figure 2. We also create two other graphs G_2 and G_3 by inserting an extra edge into G_1 in different ways. Note that the additional edge in G_3 will have a greater impact on G_1's global structure since it connects two further nodes.

We compute the normalized distances (the largest distance always equals one) between

Table 1: Normalized distance between graphs with simple perturbations.

Graph Pairs	GGD	TMD w/ NF, L = 4	TMD w/o NF, L = 4	GED
G_1, G_2	0.623	0.689	0.970	1.000
G_1, G_3	0.855	0.711	1.000	1.000
G_2, G_3	1.000	1.000	0.333	1.000

the aforementioned three graphs using different metrics (GED, TMD, and GGD) and report the results in Table 1. As observed, G_2 and G_3 have distances similar to G_1 when the TMD metric is adopted without using node features (NFs). On the other hand, the TMD metric can produce similar results as the proposed GGD metric when node features are fully utilized. Not surprisingly, the GED always produces the same distances since only one edge has been added. The above results imply that the GED and TMD (without using NFs) metrics may not properly capture the dissimilarities in the spectral properties of the graphs.

4 Computing GGDs Between Graphs of the Same Size

4.1 Phase 1: Spectral Graph Matching for Finding Node Correspondence

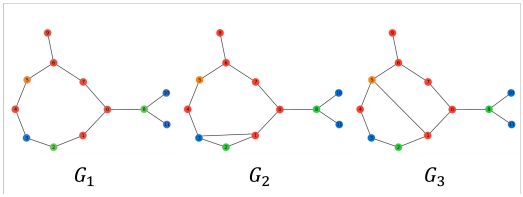

Figure 2: Graphs with simple perturbations

Computing the GGD metric between two input graphs requires solving a graph-matching problem in advance, to ensure the minimum possible distance between modified Laplacian matrices. In this work, we aim to find the infimum between two SPD matrices on the Riemannian manifold, which can be accomplished through a graph-matching phase. Graph matching techniques can be used to establish node-to-node correspondence by seeking a bijection between node sets to maximize the alignment of edge sets (Livi & Rizzi, 2013; Emmert-Streib et al., 2016; Caetano et al., 2009). This combinatorial optimization problem can be cast into a Quadratic Assignment Problem, which is NP-hard to solve or approximate (Fan et al., 2020; Wang et al., 2020).

In this study, we exploit a spectral graph matching method called GRAMPA (**GRA**ph **M**atching by **P**airwise eigen-**A**lignments) (Fan et al., 2020) to find the approximate node correspondence between two graphs. GRAMPA starts with comparing the eigenvectors of the adjacency matrices of the input graphs. Instead of comparing only the eigenvectors corresponding to the largest eigenvalues, it considers all pairs of eigenvectors/eigenvalues to generate a similarity matrix. This similarity matrix can be constructed by summing up the outer products of eigenvector pairs, weighted by a Cauchy kernel (Fan et al., 2020). Subsequently, a rounding procedure will be performed to determine the optimal match between nodes employing the similarity matrix.

Definition 4.1 (Similarity Matrix)**.** *Let G_1 and G_2 be two undirected graphs with n nodes, and let their weighted adjacency matrices be A_1 and A_2, respectively. The spectral decompositions of A_1 and A_2 are expressed as follows: $A_1 = \sum_{i=1}^{n} \zeta_i u_i u_i^\top$ and $A_2 = \sum_{j=1}^{n} \mu_j v_j v_j^\top$, where the eigenvalues are ordered such that $\zeta_1 \geq \ldots \geq \zeta_n$ and $\mu_1 \geq \ldots \geq \mu_n$. The similarity matrix $\widehat{X} \in \mathbb{R}^{n \times n}$ is defined as:*

$$\widehat{X} = \sum_{i,j=1}^{n} w\left(\zeta_i, \mu_j\right) \cdot u_i u_i^\top \mathbf{J} v_j v_j^\top, \quad w(x,y) = \frac{1}{(x-y)^2 + \eta^2}. \tag{3}$$

Here, $\mathbf{J} \in \mathbb{R}^{n \times n}$ denotes an all-one matrix and w is the Cauchy kernel of bandwidth η.

The permutation estimate matrix $\widehat{\pi}$ can be obtained by rounding $\widehat{X}$, typically achieved by solving the Linear Assignment Problem (LAP):

$$\widehat{\pi} = \operatorname{argmax} \sum_{i=1}^{n} \widehat{X}_{i,\pi(i)}, \tag{4}$$

which can be efficiently solved using the Hungarian algorithm (Fan et al., 2020). However, one simpler rounding procedure was advocated in (Fan et al., 2020) with theoretical results supporting the rounding procedure, which is given by the following equation:

$$\widehat{\pi}(i) = \operatorname*{argmax}_j \widehat{X}_{ij}, \tag{5}$$

here the permutation estimate matrix is constructed by selecting the largest index from each row. While LAP provides optimal matching, its computational complexity can become expensive for very large graphs. By carefully choosing η, the same match recovery holds if rounding is performed using equation 4 instead of solving the LAP in equation 5 (Fan et al., 2020).

Lemma 4.2 (Graph Matching Recovery). *Given symmetric matrices A_1, A_2 and Z from the Gaussian Wigner model, where $A_{2\pi^*} = A_1 + \sigma Z$, there exist constants $c, c' > 0$ such that if $1/n^{0.1} \leq \eta \leq c/\log n$ and $\sigma \leq c'\eta$, then with probability at least $1 - n^{-4}$, GRAMPA Algorithm correctly recovers the permutation matrix π^* from the Similarity matrix $\widehat{X}$ (Fan et al., 2020). Its proof can be found in the supporting documents A.8.*

Once $\widehat{\pi}$ is obtained, the best-matched mirrors of the input graphs are:

$$\text{Best Match to } A_2 = \widehat{\pi} A_1 \widehat{\pi}^\top, \quad \text{Best Match to } A_1 = \widehat{\pi}^\top A_2 \widehat{\pi} \tag{6}$$

In practice, the graph matching performance is not too sensitive to the choice of tuning parameter η. For small-sized graphs, such as the MUTAG dataset(Morris et al., 2020), setting $\eta = 0.5$ yields satisfactory results in matching. In A.16, the effect of η for computing GGDs has been comprehensively analyzed.

4.2 Phase 2: Computing Geodesic Distances Between Graph Laplacians

The GGD metric can be formally defined as the infimum length of geodesics connecting two data points in the Riemannian manifold formed by the cone of the modified graph Laplacian matrices (Lim et al., 2019). This distance metric can be imagined as a matrix representation of the geometric distance $|\log(a/b)|$ between two positive numbers a, b (Bonnabel & Sepulchre, 2010; Shamai & Kimmel, 2017; Owen & Provan, 2010).

Definition 4.3 (Graph Geodesic Distance). *Let $\mathcal{L}_1$ and $\mathcal{L}_2 \in \mathbb{S}^n_{++}$ denote two modified Laplacian matrices corresponding to two matched graphs G_1 and G_2 both having n nodes, then their Graph Geodesic Distance denoted by $GGD(G_1, G_2) : \mathbb{S}^n_{++} \times \mathbb{S}^n_{++} \to \mathbb{R}_+$, is defined as:*

$$GGD(G_1, G_2) = \left[\sum_{i=1}^{n} \log^2(\lambda_i(\mathcal{L}_1^{-1}\mathcal{L}_2)) \right]^{1/2}, \tag{7}$$

where λ_i are the generalized eigenvalues computed with the matrix pencil ($\mathcal{L}_1$, $\mathcal{L}_2$).

The above GGD formulation for computing distances between SPD matrices is based on an Affine-Invariant Riemannian Metric (AIRM) (Lim et al., 2019), while another well-known metric, the Log-Euclidean Riemannian Metric (LERM) (Ilea et al., 2018; Thanwerdas & Pennec, 2023; Chen et al., 2024) is also discussed in Appendix A.14.

4.3 Connection between GGD and Graph Structural Mismatches

Consider two graphs, G_1 and G_2, that have the same node set V, with a known correspondence between their nodes. Let L_1 and L_2 be the Laplacian matrices of these graphs, respectively. Suppose we take a subset of nodes, denoted by S and its complement, S'. We assign the value 1 to the nodes in S and the value 0 to those in S'. This defines the set S as:

$$S \stackrel{\text{def}}{=} \{v \in V : x(v) = 1\}.$$

For graph G_1, the cut for the node subset S (which is the number of edges that cross between S and S') can be computed as: $\text{cut}_{G_1}(S, S') = x^T L_1 x$.

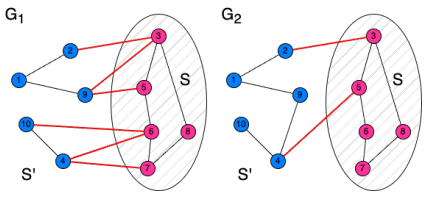

Figure 3: The cut mismatch (for the node set S) between two graphs is $\frac{6}{2} = 3$

As shown in Figure 3, for a node subset S, six edges cross between S and S' in graph G_1, whereas two edges cross in graph G_2. This ratio of edge counts in the two graphs is referred as a cut mismatch. The relationship between this cut mismatch and the generalized eigenvalue problem for the matrix pair (L_1, L_2) can be formalized using the Generalized Courant-Fischer Minimax Theorem (Golub & Van Loan, 2013; Feng, 2020).

Lemma 4.4 (The Generalized Courant-Fischer Minimax Theorem)**.** *Given two Laplacian matrices $L_1, L_2 \in \mathbb{R}^{n\times n}$ such that* $\text{null}\,(L_2) \subseteq \text{null}\,(L_1)$*, the k-th largest generalized eigenvalue of L_1 and L_2 can be computed as follows for* $1 \leq k \leq \text{rank}\,(L_2)$*:*

$$\lambda_k = \min_{\substack{\dim(U)=k \\ U\perp \text{null}(L_2)}} \max_{x\in U} \frac{x^\top L_1 x}{x^\top L_2 x}. \tag{8}$$

This theorem provides a method for bounding the maximum cut mismatch between two graphs by calculating the largest generalized eigenvalue. Specifically, we can use the following optimization problem to compute the dominant eigenvalue λ_{max} (Feng, 2020):

$$\lambda_{\max} = \max_{\substack{|x|\neq 0 \\ x^\top \mathbb{1}=0}} \frac{x^\top L_1 x}{x^\top L_2 x} \geq \max_{\substack{|x|\neq 0 \\ x(v)\in\{0,1\}}} \frac{x^\top L_1 x}{x^\top L_2 x} = \max \frac{\text{cut}_{G_1}(S, S')}{\text{cut}_{G_2}(S, S')} \tag{9}$$

From equation (9), we can see that the dominant generalized eigenvalue λ_{max} corresponds to the most significant cut mismatch between G_1 and G_2. In particular, $\lambda_1 = \lambda_{max}$ sets an upper bound on the cut mismatch between G_1 and G_2, while $\lambda_n = \lambda_{min}$ defines the upper bound of the mismatch in the reverse direction, between G_2 and G_1. Appendix A.9 illustrates this relationship with practical examples. Additionally, we illustrate the relationship between the approximate GGD values with extreme eigenvalues, compared with the accurate GGD values in Appendix A.12.

5 Computing GGDs for Graphs with Different Sizes

Submatrix selection methods To calculate geodesic distances between SPD matrices of different sizes, prior studies have proposed a submatrix adaptation method (Lim et al., 2019). In this approach, a principle submatrix with the same size as the smaller matrix is obtained from the larger matrix (Ye & Lim, 2016), and subsequently used to calculate the GGD. Furthermore, this method can be extended to project the smaller matrix into a larger one with the same size as the larger matrix (Lim et al., 2019). While these methods are efficient for handling SPD matrices, for our application taking the submatrix of the modified Laplacian can lose important nodes/edges, compromising critical graph structural properties.

Graph coarsening methods Spectral graph coarsening is a widely adopted process (Loukas, 2019; Aghdaei & Feng, 2022) for reducing graph sizes while preserving key spectral (structural) properties, such as the Laplacian eigenvalues/eigenvectors. Recent spectral graph coarsening methods aim to decompose an input graph into many distinct node clusters, so that a reduced graph can be formed by treating each node cluster as a new node, with a goal of assuring that the reduced graph will approximately retain the original graph's structure (Loukas, 2019; Han et al., 2024; Aghdaei & Feng, 2022). Therefore, when computing GGDs for graphs of different sizes, we can first adopt spectral graph coarsening to transform the bigger graph into a smaller one, so that our framework in Section 4 can be subsequently utilized. However, existing state-of-the-art graph coarsening methods do not allow us to precisely control the size of the reduced graphs.

5.1 Our Approach: Spectral Graph Coarsening by Effective Resistances

In this work, we exploit a spectral graph coarsening method leveraging effective-resistance clustering (Aghdaei & Feng, 2022), specifically designed to reduce graph size while preserving key spectral characteristics. Unlike prior work (Lim et al., 2019), which addresses dimension mismatch directly in SPD space, our coarsening method operates at the structural level before transformation into SPD matrices, making it more suitable for computing distances between graphs with unequal sizes. Our approach begins by estimating the effective resistances of all edges in the original graph. If node features are available, we also incorporate feature differences as an additional parameter. During coarsening, edges are ranked by resistance distance, and only the top few edges with the smallest effective resistances are merged into new nodes. This strategy enables precise control over the size of the reduced graph while preserving crucial spectral properties, essential for the subsequent spectral graph matching step (Phase 1 in Section 4.1).

Consider a connected, weighted, undirected graph $G = (V, E, w)$ with $|V| = n$. The effective resistance between nodes $(p, q) \in V$ plays a crucial role in various graph analysis tasks including

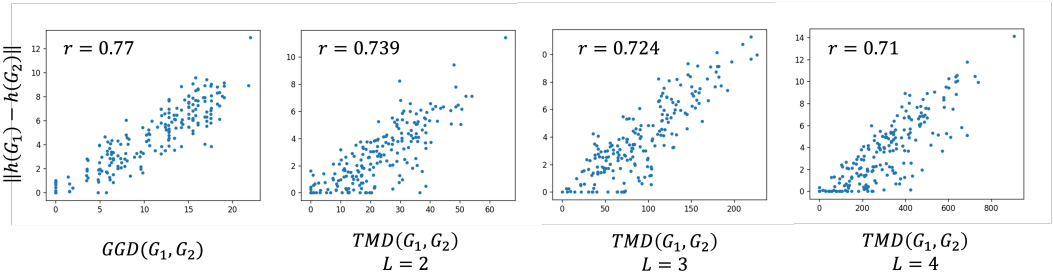

Figure 4: Correlation between graph distance metrics and GNN model outputs.

spectral sparsification algorithms (Spielman & Teng, 2011). The effective resistance distances can be accurately computed using the equation:

$$R_{eff}(p,q) = \sum_{i=2}^{n} \frac{(u_i^\top b_{pq})^2}{\sigma_i}, \tag{10}$$

where $b_p \in \mathbb{R}^n$ denote the standard basis vector with all zero entries except for the p-th entry being 1, and $b_{pq} = b_p - b_q$. $u_i \in \mathbb{R}^n$ for $i = 1, \ldots, n$ denote the unit-length, mutually-orthogonal eigenvectors corresponding to Laplacian eigenvalues σ_i for $i = 1, \ldots, n$. Background on effective resistance is presented in Appendix A.5, with its estimation detailed in Appendix A.6.

6 GGD as a Distance Metric

Assuming the graph matching problem can always find the exact correspondence between nodes, we prove that the GGD metric (based on AIRM) between any two nonempty graphs is a metric that satisfies the following conditions (Detailed proofs these properties are provided in Appendix A.7.):

- The distance between a graph and itself or between two isomorphic graphs is zero: $GGD(G, G) = 0$.
- (Positivity) The distance between two distinct graphs is positive: $GGD(G_1, G_2) \geq 0$.
- (Symmetry) The distance between G_1 and G_2 is the same of the one between G_2 and G_1: $GGD(G_1, G_2) = GGD(G_2, G_1)$.
- The triangle inequality: $GGD(G_1, G_3) \leq GGD(G_1, G_2) + GGD(G_2, G_3)$.

7 Experiments

7.1 Application of GGDs in GNN Stability Analysis

To analyze the stability of GNN models (Bronstein et al., 2017; Garg et al., 2020; Duvenaud et al., 2015), we conducted multiple experiments with the GGD and TMD metrics. GNNs typically operate by a message-passing mechanism (Gilmer et al., 2017), where at each layer, nodes send their feature representations to their neighbors. The feature representation of each node is initialized to its original features and is updated by repeatedly aggregating incoming messages from neighbors. In our experiment, we relate GGD to the Graph Isomorphism Networks (GIN) (Xu et al., 2019), one of the most widely applied and powerful GNNs, utilizing the MUTAG dataset (Morris et al., 2020) as our reference graph dataset. The objective is to analyze the relationship between the input distance $GGD(G_1, G_2)$ and the distance between the output GIN vectors, $\|h(G_1) - h(G_2)\|$ for randomly selected pairs of graphs.

We employed a three-layer GIN network as described in (Xu et al., 2019). This network uses GIN convolutional layers to update tensors of nodes based on their neighboring nodes and then aggregates those outputs in a vector representation, followed by linear layers for classification tasks. Thus it outputs a single vector $h(G)$ for the entire graph G. The result is illustrated in Figure 4.

We observe a strong correlation between GGD and the output distance, as indicated by a high Pearson correlation coefficient. This finding implies the effectiveness of the proposed GGD metric for analyzing the stability of GNN models (Chuang & Jegelka, 2022). To compare GGD with existing

Table 2: Classification accuracies for various models on graph datasets.

DATASET	MUTAG	PC-3H	SW-620H	BZR
GGD	86.24±7.89	**78.34±1.60**	**77.6±3.50**	**83.23±6.25**
TMD, L = 2	76.19±5.26	–	–	–
TMD, L = 3	77.34±5.26	71.24±2.45	70.22±2.29	73.43±2.44
TMD, L = 4	78.20±5.26	71.37±1.42	70.84±2.29	73.96±4.88
TMD, L = 5	78.20±5.26	71.89±2.40	71.20±1.88	75.13±2.44
GCN (KIPF & WELLING, 2017)	77.37±3.95	70.56±1.66	69.44±0.94	72.56±3.66
GIN (XU ET AL., 2019)	82.60±4.60	75.34±1.10	73.36±2.32	77.09±3.66
DGCNN (ZHANG ET AL., 2018)	76.66±3.19	73.79±0.75	74.37±1.54	72.38±1.08
WWL (TOGNINALLI ET AL., 2019)	72.39±2.63	65.46±1.11	68.06±0.86	72.37±1.22
WL SUBTREE (SHERVASHIDZE ET AL., 2011)	76.81±6.30	68.43±0.76	69.36±1.20	N/A
FGW (TITOUAN ET AL., 2019)	**88.33±5.26**	61.77±1.11	58.28±1.02	51.03±2.63

metrics, we repeat this experiment using TMD without considering node attributes (features). As shown in Figure 4, GGD demonstrates a better correlation with GIN outputs than the TMD metric across different levels. These findings indicate that when dealing with graphs without node features, GGD should be adopted for the stability analysis of graph learning models. The performance of GGD under partially missing node features is further discussed in Appendix A.11.

7.2 APPLICATION OF GGDS IN GRAPH CLASSIFICATION TASKS

We evaluate whether the GGD metric aligns with graph labels in graph classification tasks using datasets from TUDataset (Morris et al., 2020). We employ a Support Vector Classifier (SVC) ($C = 1$) with an indefinite kernel $e^{-\gamma * GGD(G_1,G_2)}$, which can be viewed as a noisy observation of the true positive semidefinite kernel (Luss & d'Aspremont, 2007). The parameter γ is selected through cross-validation from the set {0.01, 0.05, 0.1}. For comparative analysis with existing methods, we include graph kernels based on graph subtrees: the WL subtree kernel (Shervashidze et al., 2011); and two widely adopted GNNs: graph isomorphism network (GIN) (Xu et al., 2019) and graph convolutional networks (GCN) (Kipf & Welling, 2017).

Table 2 presents the mean and standard deviation over five independent trials with a 90%-10% train-test split. For most cases, GGD consistently outperforms the performance of state-of-the-art GNNs, graph kernels, and metrics when node attributes are missing. Additionally, we observe that GGD allows us to obtain better results for larger datasets than smaller ones.

7.3 APPLICATION OF GGDS IN DATASET DISTANCE

To extend the application of GGD beyond structured graph datasets, we explore its utility in measuring distances between datasets. In transfer learning, finding the distance between datasets helps to quantify how similar or dissimilar the source and target domains are, guiding the adaptation of knowledge from one domain to another (Alvarez-Melis & Fusi, 2020). We calculated this distance by first converting standard datasets into graph representations and then computing distances between these graphs using GGD framework.

To construct a graph from a dataset, we treat data points as nodes and establish edges based on a k-nearest neighbor (k-NN) approach. Nodes representing similar data points are connected, forming an initial dense graph. To further refine the structure, we apply a spectral graph sparsification procedure to reduce the number of edges while preserving key connectivity properties. The process of creating graphs from the dataset is further explained in Appendix A.18.

Once the graphs are obtained, we apply the GGD framework to compute pairwise distances. These distances are then compared with established dataset distance measures based on Optimal Transport (Alvarez-Melis & Fusi, 2020). We also explore how these distances correlate with dataset *Transferability*, which is the improvement in model performance when pretrained on a source dataset and fine-tuned on a target dataset. Transferability $\mathcal{T}$ of a source domain $\mathcal{D}_S$ to a target domain $\mathcal{D}_T$ is defined as the relative decrease in classification error when adapting compared to training only on the target domain (Alvarez-Melis & Fusi, 2020).

$$\mathcal{T}\left(\mathcal{D}_S \to \mathcal{D}_T\right) = 100 \times \frac{\text{error}\left(\mathcal{D}_S \to \mathcal{D}_T\right) - \text{error}\left(\mathcal{D}_T\right)}{\text{error}\left(\mathcal{D}_T\right)}. \tag{11}$$

Our results indicate that GGD-based dataset distances provide meaningful insights into dataset relationships and transfer learning ability, matching the performance of conventional distance metrics, demonstrated in Figure 5.

7.4 Runtime Complexity Analysis and Comparison

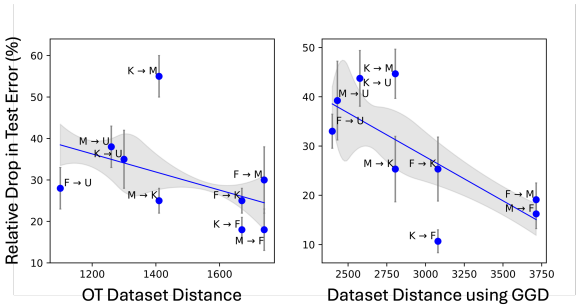

Figure 5: Dataset distance vs. Adaptation for *NIST datasets (M: MNIST, K: KMNIST, F: Fashion-MNIST, U: USPS).

When comparing various graph distance metrics, a primary consideration is their computational complexity. Conventional approaches usually require intricate computations that frequently have cubic or higher complexities. For our problem, the spectral graph matching step requires the eigenvalue decomposition of adjacency matrices and solving the linear assignment problem (LAP). Eigenvalue decomposition of an $n \times n$ matrix has a complexity of $O(n^3)$ (Borodin & Munro, 1975; Flamary et al., 2021), while solving the LAP using the Hungarian algorithm also has a runtime complexity of $O(n^3)$. Similarly, calculating the generalized eigenvalue of two SPD matrices entails a cubic complexity. Consequently, the overall complexity of GGD calculation is $O(n^3)$. On the other hand, TMD is an OT-based distance metric with a complexity of $O(n^3 \log(n))$ (Chuang & Jegelka, 2022; Flamary et al., 2021). Therefore, GGD exhibits slightly better (lower) runtime complexity than TMD.

To evaluate runtime performance, we compare GGD and TMD on both small graphs (MUTAG, BZR) and large graphs (PC-3H, SW-620H) from the TUDataset (Morris et al., 2020). Table 3 reports the average time (in seconds) to compute 100 pairwise distances, averaged over five runs. GGD consistently outperforms TMD across all datasets, especially on larger graphs with more nodes. Since TMD requires deeper levels to capture structural information, its runtime increases rapidly. In contrast, GGD is approximately 6-9 times faster, making it significantly more efficient, particularly on large graphs. More details are provided in Appendix A.20 and Appendix A.13.

Table 3: Runtime comparison for different distance metrics on various datasets.

	MUTAG	PC-3H	SW-620H	BZR
GGD	**4.87s**	**31.89s**	**45.37s**	**5.80s**
TMD, L = 3	5.29s	88.60s	98.69s	7.22s
TMD, L = 4	7.90s	112.12s	134.38s	10.34s
TMD, L = 6	11.27s	273.31s	287.92s	14.98s

8 Conclusion

In this work, we introduce Graph Geodesic Distance (GGD), a novel spectral graph distance metric based on graph matching and the infimum on a Riemannian manifold. GGD captures key structural mismatches crucial for graph classification tasks, and we demonstrate its effectiveness for analyzing GNN model stability as well as for graph classification, delivering superior performance even with partial node features. Additionally, GGD can evaluate transferability by calculating distances between datasets. Despite some limitations (Appendix A.19), GGD offers a principled and practical tool for comparing graphs across tasks and domains.

LLM Usage

We used LLM-based tools to improve the clarity of a few sentences and to correct grammatical errors.

REFERENCES

Ali Aghdaei and Zhuo Feng. Hyperef: Spectral hypergraph coarsening by effective-resistance clustering. In *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, pp. 1–9, 2022.

David Alvarez-Melis and Nicolo Fusi. Geometric dataset distances via optimal transport. *Advances in Neural Information Processing Systems*, 33:21428–21439, 2020.

John Adrian Bondy, Uppaluri Siva Ramachandra Murty, et al. *Graph theory with applications*, volume 290. Macmillan London, 1976.

Silvere Bonnabel and Rodolphe Sepulchre. Riemannian metric and geometric mean for positive semidefinite matrices of fixed rank. *SIAM Journal on Matrix Analysis and Applications*, 31(3): 1055–1070, 2010.

Karsten Borgwardt, Elisabetta Ghisu, Felipe Llinares-López, Leslie O'Bray, and Bastian Rieck. Graph kernels: State-of-the-art and future challenges. *Foundations and Trends® in Machine Learning*, 13(5-6):531–712, 2020. ISSN 1935-8237. doi: 10.1561/2200000076.

A. Borodin and I. Munro. *The Computational Complexity of Algebraic and Numeric Problems*. Elsevier Computer Science Library, Theory of Computation Series. American Elsevier Publishing Company, 1975. ISBN 9780608163741.

Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

Horst Bunke and Kim Shearer. A graph distance metric based on the maximal common subgraph. *Pattern recognition letters*, 19(3-4):255–259, 1998.

Tibério S Caetano, Julian J McAuley, Li Cheng, Quoc V Le, and Alex J Smola. Learning graph matching. *IEEE transactions on pattern analysis and machine intelligence*, 31(6):1048–1058, 2009.

Laetitia Chapel, Mokhtar Z Alaya, and Gilles Gasso. Partial optimal tranport with applications on positive-unlabeled learning. *Advances in Neural Information Processing Systems*, 33:2903–2913, 2020.

Xu Chen, Siheng Chen, Jiangchao Yao, Huangjie Zheng, Ya Zhang, and Ivor W Tsang. Learning on attribute-missing graphs. *IEEE transactions on pattern analysis and machine intelligence*, 44(2): 740–757, 2022.

Ziheng Chen, Yue Song, Tianyang Xu, Zhiwu Huang, Xiao-Jun Wu, and Nicu Sebe. Adaptive log-euclidean metrics for spd matrix learning. *IEEE Transactions on Image Processing*, 2024.

Wuxinlin Cheng, Chenhui Deng, Ali Aghdaei, Zhiru Zhang, and Zhuo Feng. Sagman: Stability analysis of graph neural networks on the manifolds. *arXiv preprint arXiv:2402.08653*, 2024.

Timothy Chu, Yu Gao, Richard Peng, Sushant Sachdeva, Saurabh Sawlani, and Junxing Wang. Graph sparsification, spectral sketches, and faster resistance computation via short cycle decompositions. *SIAM Journal on Computing*, 52(6):FOCS18–85, 2020.

Ching-Yao Chuang and Stefanie Jegelka. Tree mover's distance: Bridging graph metrics and stability of graph neural networks. *Advances in Neural Information Processing Systems*, 35:2944–2957, 2022.

Ching-Yao Chuang, Antonio Torralba, and Stefanie Jegelka. Estimating generalization under distribution shifts via domain-invariant representations. *International conference on machine learning*, 2020.

Fan RK Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.

Keenan Crane, Marco Livesu, Enrico Puppo, and Yipeng Qin. A survey of algorithms for geodesic paths and distances. *arXiv preprint arXiv:2007.10430*, 2020.

David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems*, 28, 2015.

Wendy Ellens, Floske M Spieksma, Piet Van Mieghem, Almerima Jamakovic, and Robert E Kooij. Effective graph resistance. *Linear algebra and its applications*, 435(10):2491–2506, 2011.

Frank Emmert-Streib, Matthias Dehmer, and Yongtang Shi. Fifty years of graph matching, network alignment and network comparison. *Information sciences*, 346:180–197, 2016.

Zhou Fan, Cheng Mao, Yihong Wu, and Jiaming Xu. Spectral graph matching and regularized quadratic relaxations i: The gaussian model. *arXiv preprint arXiv:1907.08880*, 2019.

Zhou Fan, Cheng Mao, Yihong Wu, and Jiaming Xu. Spectral graph matching and regularized quadratic relaxations: Algorithm and theory. In *International conference on machine learning*, pp. 2985–2995. PMLR, 2020.

Zhuo Feng. Grass: Graph spectral sparsification leveraging scalable spectral perturbation analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 39(12):4944–4957, 2020.

Zhuo Feng. Sgl: Spectral graph learning from measurements. In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pp. 727–732. IEEE, 2021.

Mirtha-Lina Fernández and Gabriel Valiente. A graph distance metric combining maximum common subgraph and minimum common supergraph. *Pattern Recognition Letters*, 22(6-7):753–758, 2001.

Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, et al. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021.

Xinbo Gao, Bing Xiao, Dacheng Tao, and Xuelong Li. A survey of graph edit distance. *Pattern Analysis and applications*, 13:113–129, 2010.

Vikas Garg, Stefanie Jegelka, and Tommi Jaakkola. Generalization and representational limits of graph neural networks. In *International Conference on Machine Learning*, pp. 3419–3430. PMLR, 2020.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pp. 1263–1272. PMLR, 2017.

Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013.

Xiaoxue Han, Zhuo Feng, and Yue Ning. A topology-aware graph coarsening framework for continual graph learning. *arXiv preprint arXiv:2401.03077*, 2024.

Jonas MB Haslbeck and Lourens J Waldorp. How well do network models predict observations? on the importance of predictability in network models. *Behavior research methods*, 50:853–861, 2018.

David Haussler et al. Convolution kernels on discrete structures. Technical report, Citeseer, 1999.

Zhiwu Huang, Ruiping Wang, Shiguang Shan, and Xilin Chen. Learning euclidean-to-riemannian metric for point-to-set classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1677–1684, 2014.

Zhiwu Huang, Ruiping Wang, Shiguang Shan, Xianqiu Li, and Xilin Chen. Log-euclidean metric learning on symmetric positive definite manifold with application to image set classification. In *International conference on machine learning*, pp. 720–729. PMLR, 2015.

Ioana Ilea, Lionel Bombrun, Salem Said, and Yannick Berthoumieu. Covariance matrices encoding based on the log-euclidean and affine invariant riemannian metrics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 393–402, 2018.

Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*, 2017.

John M Lee. *Introduction to Riemannian manifolds*, volume 2. Springer, 2018.

Tao Li, Han Dong, Yongtang Shi, and Matthias Dehmer. A comparative analysis of new graph distance measures and graph edit distance. *Information Sciences*, 403:15–21, 2017.

Jörg Liesen and Zdeněk Strakoš. Krylov subspace methods: Principles and analysis. *Krylov Subspace Methods: Principles and Analysis*, 10 2012. doi: 10.1093/acprof:oso/9780199655410.001.0001.

Lek-Heng Lim, Rodolphe Sepulchre, and Ke Ye. Geometric distance between positive definite matrices of different dimensions. *IEEE Transactions on Information Theory*, 65(9):5401–5405, 2019.

Lorenzo Livi and Antonello Rizzi. The graph matching problem. *Pattern Analysis and Applications*, 16:253–283, 2013.

Andreas Loukas. Graph reduction with spectral and cut guarantees. *J. Mach. Learn. Res.*, 20(116): 1–42, 2019.

Ronny Luss and Alexandre d'Aspremont. Support vector machine classification with indefinite kernels. *Advances in neural information processing systems*, 20, 2007.

Facundo Mémoli. Gromov–wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics*, 11:417–487, 2011.

Maher Moakher. A differential geometric approach to the geometric mean of symmetric positive-definite matrices. *SIAM journal on matrix analysis and applications*, 26(3):735–747, 2005.

Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 4602–4609, 2019.

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.

Megan Owen and J Scott Provan. A fast algorithm for computing geodesic distances in tree space. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8(1):2–13, 2010.

Karl Pearson. The problem of the random walk. *Nature*, 72(1865):294–294, 1905.

Xavier Pennec, Pierre Fillard, and Nicholas Ayache. A riemannian framework for tensor computing. *International Journal of computer vision*, 66:41–66, 2006.

Jan Ramon and Thomas Gärtner. Expressivity versus efficiency of graph kernels. In *Proceedings of the first international workshop on mining graphs, trees and sequences*, pp. 65–74, 2003.

Emanuele Rossi, Henry Kenlay, Maria I Gorinova, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M Bronstein. On the unreasonable effectiveness of feature propagation in learning on graphs with missing node features. In *Learning on Graphs Conference*, pp. 11–1. PMLR, 2022.

Sushmita Roy, Terran Lane, and Margaret Werner-Washburne. Learning structurally consistent undirected probabilistic graphical models. In *Proceedings of the 26th annual international conference on machine learning*, pp. 905–912, 2009.

Yousef Saad. *Numerical methods for large eigenvalue problems: revised edition*. SIAM, 2011.

Alberto Sanfeliu and King-Sun Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(3):353–362, 1983. doi: 10.1109/TSMC.1983.6313167.

Gil Shamai and Ron Kimmel. Geodesic distance descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6410–6418, 2017.

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.

Daniel A Spielman and Shang-Hua Teng. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4):981–1025, 2011.

Edric Tam and David Dunson. Multiscale graph comparison via the embedded laplacian discrepancy. *arXiv preprint arXiv:2201.12064*, 2022.

Yann Thanwerdas and Xavier Pennec. O (n)-invariant riemannian metrics on spd matrices. *Linear Algebra and its Applications*, 661:163–201, 2023.

Vayer Titouan, Nicolas Courty, Romain Tavenard, and Rémi Flamary. Optimal transport for structured data with application on graphs. In *International Conference on Machine Learning*, pp. 6275–6284. PMLR, 2019.

Matteo Togninalli, Elisabetta Ghisu, Felipe Llinares-López, Bastian Rieck, and Karsten Borgwardt. Wasserstein weisfeiler-lehman graph kernels. *Advances in neural information processing systems*, 32, 2019.

Raviteja Vemulapalli and David W Jacobs. Riemannian metric learning for symmetric positive definite matrices. *arXiv preprint arXiv:1501.02393*, 2015.

S Vichy N Vishwanathan, Nicol N Schraudolph, Risi Kondor, and Karsten M Borgwardt. Graph kernels. *The Journal of Machine Learning Research*, 11:1201–1242, 2010.

Tao Wang, He Liu, Yidong Li, Yi Jin, Xiaohui Hou, and Haibin Ling. Learning combinatorial solver for graph matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 7568–7577, 2020.

Boris Weisfeiler and Andrei Leman. The reduction of a graph to canonical form and the algebra which appears therein. *nti, Series*, 2(9):12–16, 1968.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

Yunwen Xu, Srinivasa M Salapaka, and Carolyn L Beck. A distance metric between directed weighted graphs. In *52nd IEEE Conference on Decision and Control*, pp. 6359–6364. IEEE, 2013.

Ke Ye and Lek-Heng Lim. Schubert varieties and distances between subspaces of different dimensions. *SIAM Journal on Matrix Analysis and Applications*, 37(3):1176–1197, 2016.

Kisung You and Hae-Jeong Park. Re-visiting riemannian geometry of symmetric positive definite matrices for the analysis of functional connectivity. *NeuroImage*, 225:117464, 2021.

Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. An end-to-end deep learning architecture for graph classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Yuehua Zhu, Muli Yang, Cheng Deng, and Wei Liu. Fewer is more: A deep graph metric learning perspective using fewer proxies. *Advances in Neural Information Processing Systems*, 33:17792–17803, 2020.

A Appendix

A.1 Algorithm Flow

Algorithm 1 GGD: Geodesic Graph Distance

Input: Graphs $G_1 = (V_1, E_1, w_1)$, $G_2 = (V_2, E_2, w_2)$, tuning parameter $\eta > 0$, small diagonal value $0 < \epsilon \ll 1$, node feature weight α
Output: GGD Value
Compute the adjacency matrices A_1, A_2
if shape$(A_1) \neq$ shape(A_2) **then**
    Assign the larger graph to G_1, and the smaller graph to G_2
    while shape$(A_1) \geq$ shape(A_2) **do**
        Compute the effective resistance $R_{\text{eff}}(p, q)$ of each edge $(p, q) \in E_1$
        Compute the modified effective resistance $R^*_{\text{eff}}(p, q) = R_{\text{eff}}(p, q) + \alpha \|NF_p - NF_q\|$
        Coarsen the edge with the lowest $R^*_{\text{eff}}(p, q)$
        Update A_1
    end while
end if
Compute eigenvectors u_i, v_i and eigenvalues ζ_i, μ_i of A_1 and A_2, respectively
Compute the similarity matrix $\hat{X} \in \mathbb{R}^{n \times n}$
Solve Linear Assignment Problem to compute the permutation estimate matrix $\hat{\pi}$
Update A_2 by multiplying with $\hat{\pi}$ to get best match with A_1
Derive L_1 and L_2 from A_1 and A_2
Add ϵ to diagonal values of L_1 and L_2
Compute GGD value using Equation 7
return GGD

A.2 Tree Mover's Distance

By progressively adding neighboring nodes of the graph, to the previous node at each level, one obtains the computation tree of a node. These tree structures are crucial in graph analysis (Weisfeiler & Leman, 1968; Pearson, 1905) and graph kernels (Ramon & Gärtner, 2003; Shervashidze et al., 2011). TMD uses hierarchical optimal transport (OT) to analyze these computational trees from input graphs. For a graph $G = (V, E)$ with node features $f_v \in \mathbb{R}^s$ for node $v \in V$, let $T^1_v = v$, and T^L_v be the depth-L computation tree of node v. The multiset of these trees for G is $T^L_G = \{T^L_v\}_{v \in V}$. The number and shape of trees must match to calculate optimal transport between two multisets of trees. If multisets are uneven, they are augmented with blank nodes. For multisets T_p and T_q, the augmenting function σ adds blank trees to equalize their sizes. A blank tree $T_{\mathbb{O}}$ has a single node with a zero vector feature $\mathbb{O}_p \in \mathbb{R}^s$:

$$\sigma(T_p, T_q) \to \left(T_p \cup T_{\mathbb{O}}^{\max(|T_q|-|T_p|,0)},\ T_q \cup T_{\mathbb{O}}^{\max(|T_p|-|T_q|,0)}\right) \tag{12}$$

Let $X = \{x_i\}_{i=1}^k$ and $Y = \{y_i\}_{j=1}^k$ be two data multisets and $C \in \mathbb{R}^{k \times k}$ be the transportation cost for each data pair: $C_{ij} = d(x_i, y_j)$, where d is the distance between x_i and y_j. The unnormalized Optimal Transport between X and Y is defined as follows:

$$\begin{aligned} &\mathrm{OT}_d(X, Y) := \min_{\gamma \in \Gamma(X,Y)} \langle C, \gamma \rangle \\ &\Gamma(X, Y) = \left\{\gamma \in \mathbb{R}_+^{m \times m} \mid \gamma \mathbb{1}_m = \gamma^\top \mathbb{1}_m = \mathbb{1}_m\right\}. \end{aligned} \tag{13}$$

Here Γ is the set of transportation plans that satisfies the flow constrain $\gamma \mathbb{1}_m = \gamma^\top \mathbb{1}_m = \mathbb{1}_m$. (Chuang & Jegelka, 2022). Now, the distance between two trees T_p and T_q with roots r_p and r_q is defined recursively:

$$\mathrm{TD}_w(T_p, T_q) := \begin{cases} \left\|f_{r_p} - f_{r_q}\right\| + w(L) \cdot \mathrm{OT}_{\mathrm{TD}_w}\left(\sigma\left(T_{r_p}, T_{r_q}\right)\right), \text{if } L > 1 \\ \left\|f_{r_p} - f_{r_q}\right\|, \quad \text{otherwise} \end{cases} \tag{14}$$

where L is the maximum depth of T_p and T_q, and w is a depth-dependent weighting function. Subsequently, the concept of distance from individual trees is extended to entire graphs. For graphs G_1 and G_2, multisets $\mathbf{T}^L_{G_1}$ and $\mathbf{T}^L_{G_2}$ of depth-L computation trees are used to calculate the TMD value using equation 2).

A.3 Graph Adjacency and Laplacian Matrices

For an undirected graph $G = (V, E, w)$, where V represents the set of nodes (vertices), E represents the set of edges, and w denotes the associated edge weights, the adjacency matrix A is defined as follows:

$$A(i,j) = \begin{cases} w(i,j), & \text{if } (i,j) \in E. \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

Let D denote the diagonal matrix where $D(i,i)$ is equal to the (weighted) degree of node i. The graph Laplacian matrix is then given by $L = D - A$. The rank of the Laplacian matrix of a graph G is $n - c(G)$, where n is the number of nodes and $c(G)$ is the number of connected components in the graph. For a connected graph, this implies that the rank of the Laplacian matrix is $n - 1$, meaning Laplacian matrices are not full-rank (Bondy et al., 1976).

A.4 Riemannian Manifold

A manifold is a type of topological space that resembles Euclidean space in small, local regions around each point. In other words, for every point on a manifold, there is a neighborhood that is similar to a flat multidimensional space. A Riemannian manifold is a type of manifold equipped with a smoothly varying inner product on the tangent spaces at each point. This means that for every point on the manifold, the tangent space has a way of measuring distances and angles, and these measurements change smoothly from point to point (Lee, 2018). In simpler terms, a Riemannian manifold is a smooth, curved space that locally behaves like Euclidean space but has its own geometric properties, such as how distances, angles, and volumes are defined. These properties are determined by a Riemannian metric, which generalizes the concept of measuring lengths and angles in flat space to curved spaces (Lee, 2018).

A Riemannian manifold can have curvature, unlike a flat space. This curvature allows the study of geometric shapes ranging from spheres and cylinders to more abstract surfaces. The Riemannian structure enables us to compute geodesics, volumes, and various types of curvature. This makes Riemannian manifolds fundamental in fields like differential geometry and physics, and increasingly important in data science, where curved spaces are used to model complex datasets (You & Park, 2021).

A.5 Effective Resistance in Graph Theory

Effective resistance, also known as resistance distance, is a concept in spectral graph theory that draws an analogy between electrical networks and graphs, helping to quantify how easily current can flow between two nodes, where the edges are treated as resistors. The effective resistance between nodes provides insight into the connectivity between the network. This means two nodes with lower effective resistance values have higher connectivity (Ellens et al., 2011).

A.6 Scalable estimation of effective resistances

To address the computational complexity associated with directly computing eigenvalues and eigenvectors required for estimating edge effective resistances, we leverage a scalable framework for approximating the eigenvectors of the graph Laplacian matrix using the Krylov subspace (Saad, 2011). Let A denote the adjacency matrix of a graph G, consider its order-m Krylov subspace $\mathbf{K}_m(A, x)$ that is a vector space spanned by the vectors computed through power iterations $x, Ax, A^2x, \ldots, A^{m-1}x$ (Liesen & Strakoš, 2012). By enforcing orthogonality among the above vectors in the Krylov subspace, a new set of mutually orthogonal vectors of unit lengths can be constructed for approximating the original Laplacian eigenvectors in 10, which are denoted as $\tilde{u}_1$, $\tilde{u}_2$, …, $\tilde{u}_m$. To estimate the effective resistance between two nodes p and q, we can exploit the approximated eigenvectors:

$$R_{eff}(p,q) \approx \sum_{i=1}^{m} \frac{(\tilde{u}_i^\top b_{pq})^2}{\tilde{u}_i^\top L \tilde{u}_i}, \tag{16}$$

where $\tilde{u}_i$ represents the approximated eigenvector corresponding to the i-th eigenvalue of L.

Graph coarsening with node features In order to account for the variation in node features along with edge resistive distance, we can use the following modified effective resistance formulation:

$$R^*_{eff}(p,q) = R_{eff}(p,q) + \alpha\|f_p - f_q\|, \tag{17}$$

where f_p and f_q are node feature vectors of nodes p and q, respectively, while α is a weighting factor that determines the effect of node feature information in the graph coarsening process. For instance, if the weight is sufficiently large, the modified effective resistance between nodes with different features will always exceed that of nodes with similar features, effectively preventing their coarsening.

A.7 Detailed Proofs Showing GGD is a Metric

A.7.1 Identity Property

Proof. Let the corresponding SPD matrix of the graph G be $\mathcal{L} \in \mathbb{S}^n_{++}$. From Equation 7, we have:

$$GGD(G,G) = \left[\sum_{i=1}^{n} \log^2(\lambda_i(\mathcal{L}^{-1}\mathcal{L}))\right]^{1/2} = \left[\sum_{i=1}^{n} \log^2(\lambda_i(I))\right]^{1/2}.$$

The identity matrix has only one eigenvalue, which is 1. So, $GGD(G,G) = \left[\log^2(1)\right]^{1/2} = 0$. □

A.7.2 Positivity Property

Proof. Let the corresponding SPD matrices of the graphs G_1 and G_2 be $\mathcal{L}_1$, $\mathcal{L}_2 \in \mathbb{S}^n_{++}$. Let the generalized eigenvalues of $(\mathcal{L}_1^{-1}\mathcal{L}_2)$ be $\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_n$. From Equation 7, we get:

$$GGD(G_1,G_2) = \left[\log^2(\lambda_1) + \log^2(\lambda_2) + \log^2(\lambda_3) + \ldots + \log^2(\lambda_n)\right]^{1/2}.$$

Now, $\log^2(\lambda_1) + \log^2(\lambda_2) + \log^2(\lambda_3) + \ldots + \log^2(\lambda_n) \geq 0$, for any values of λ_i.

We can conclude, $GGD(G_1,G_2) \geq 0$. □

A.7.3 Symmetry Property

Proof. Let the corresponding SPD matrices of the graphs G_1 and G_2 be $\mathcal{L}_1$, $\mathcal{L}_2 \in \mathbb{S}^n_{++}$. Let the generalized eigenvalues of $(\mathcal{L}_1^{-1}\mathcal{L}_2)$ be $\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_n$. From Equation 7, we get:

$$GGD(G_1,G_2) = \left[\sum_{i=1}^{n} \log^2(\lambda_i)\right]^{1/2}.$$

Suppose λ_i is an eigenvalue of $\mathcal{L}_1^{-1}\mathcal{L}_2$ with corresponding eigenvector v_i, i.e.,

$$\mathcal{L}_1^{-1}\mathcal{L}_2 v_i = \lambda_i v_i.$$

Multiplying both sides by $\mathcal{L}_1$, we get:

$$\mathcal{L}_2 v_i = \lambda_i \mathcal{L}_1 v_i.$$

Now multiply both sides by $\mathcal{L}_2^{-1}$:

$$v_i = \lambda_i \mathcal{L}_2^{-1}\mathcal{L}_1 v_i.$$

Rearranging, we obtain:

$$\mathcal{L}_2^{-1}\mathcal{L}_1 v_i = \frac{1}{\lambda_i} v_i.$$

So, the eigenvalues of $\mathcal{L}_2^{-1}\mathcal{L}_1$ are $\frac{1}{\lambda_1}, \frac{1}{\lambda_2}, \ldots, \frac{1}{\lambda_n}$, with the same eigenvectors.

$$GGD(G_2,G_1) = \left[\sum_{i=1}^{n} \log^2(\frac{1}{\lambda_i})\right]^{1/2}.$$

Now, $\log\left(\frac{1}{\lambda_i}\right) = -\log(\lambda_i)$; so, $\log^2\left(\frac{1}{\lambda_i}\right) = \log^2(\lambda_i)$.

So, we can conclude $GGD(G_1,G_2) = GGD(G_2,G_1)$. □

A.7.4 TRIANGLE INEQUALITY

Proof. Let, $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3 \in \mathbb{S}^n_{++}$ are three SPD matrices corresponding to graphs G_1, G_2, G_3.

Now, The Frobenius norm $\|X\|_F$ is the geodesic length at $d(\exp X, I) = \|X\|_F$ (Bonnabel & Sepulchre, 2010). Hence at identity, $d(\mathcal{L}, I) = \|\log \mathcal{L}\|_F$.

From (Bonnabel & Sepulchre, 2010; You & Park, 2021) we get,

$$GGD(G_1, G_2) = GGD\left(G_1^{-1/2} G_2 G_1^{-1/2}, I\right) = \left\|\log\left(\mathcal{L}_1^{-1/2}\mathcal{L}_2\mathcal{L}_1^{-1/2}\right)\right\|_F = \left\|\log\left(\mathcal{L}_1^{-1}\mathcal{L}_2\right)\right\|_F. \tag{18}$$

We know,

$$\mathcal{L}_1^{-1}\mathcal{L}_3 = \mathcal{L}_1^{-1}(\mathcal{L}_2\mathcal{L}_2^{-1})\mathcal{L}_3 = (\mathcal{L}_1^{-1}\mathcal{L}_2)(\mathcal{L}_2^{-1}\mathcal{L}_3).$$

Now using the Frobenius norm inequality, we get:

$$\|\mathcal{L}_1^{-1}\mathcal{L}_3\| = \|(\mathcal{L}_1^{-1}\mathcal{L}_2)(\mathcal{L}_2^{-1}\mathcal{L}_3)\| \leq \|\mathcal{L}_1^{-1}\mathcal{L}_2\|\|\mathcal{L}_2^{-1}\mathcal{L}_3\|.$$

Now taking logarithms on both sides:

$$\|\log(\mathcal{L}_1^{-1}\mathcal{L}_3)\| \leq \|\log(\mathcal{L}_1^{-1}\mathcal{L}_2)\| + \|\log(\mathcal{L}_2^{-1}\mathcal{L}_3)\|.$$

Using Equation 7, we conclude:

$$GGD(G_1, G_3) \leq GGD(G_1, G_2) + GGD(G_2, G_3).$$

□

A.8 GRAPH MATCHING RECOVERY

Given symmetric matrices A_1, A_2 and Z from the Gaussian Wigner model, where $A_{2\pi^*} = A_1 + \sigma Z$, there exist constants $c, c' > 0$ such that if $1/n^{0.1} \leq \eta \leq c/\log n$ and $\sigma \leq c'\eta$, then with probability at least $1 - n^{-4}$ for all large n, the matrix $\widehat{X}$ in equation 3 satisfies,

$$\min_{i\in[n]} \widehat{X}_{i,\pi^*(i)} > \max_{i,j\in[n]:j\neq\pi^*(i)} \widehat{X}_{ij} \tag{19}$$

and hence, the GRAMPA algorithm correctly recovers the permutation estimation matrix π^*.

Now, this proof is divided into two parts:

Lemma A.1 (Noiseless Setting Diagonal Dominance)**.** *In a noiseless situation, means replacing A_2 with A_1, similarity matrix $\widehat{X}^*$ is defined as:*

$$\widehat{X}^* = \widehat{X}(A_1, A_1) = \sum_{i,j=1}^{n} \frac{u_i u_i^T \mathbf{J} u_j u_j^T}{(\zeta_i - \zeta_j)^2 + \eta^2}. \tag{20}$$

For some constants $C, c > 0$, if $1/n^{0.1} < \eta < c/\log n$, then with probability at least $1 - 5n^{-5}$ for large n, it can be proved that the diagonal components of $\widehat{X}^*$ are dominant by showing (Fan et al., 2019):

$$\min_{i\in[n]} (\widehat{X}^*)_{ii} > \frac{1}{3\eta^2} \tag{21}$$

and

$$\max_{i,j\in[n]:i\neq j} (\widehat{X}^*)_{ij} < C\left(\frac{\sqrt{\log n}}{\eta^{3/2}} + \frac{\log n}{\eta}\right). \tag{22}$$

Lemma A.2 (Bounding the Noise Impact)**.** *The difference between the similarity matrix X in the presence of noise and the noiseless situation is bounded. If $\eta > 1/n^{0.1}$, then for a constant $C > 0$, with probability at least $1 - 2n^{-5}$ for large n, it can be shown (Fan et al., 2019):*

$$\max_{i,j\in[n]} |\widehat{X}_{ij} - (\widehat{X}^*)_{ij}| < C\sigma\left(\frac{1}{\eta^3} + \frac{\log n}{\eta^2}\left(1 + \frac{\sigma}{\eta}\right)\right). \tag{23}$$

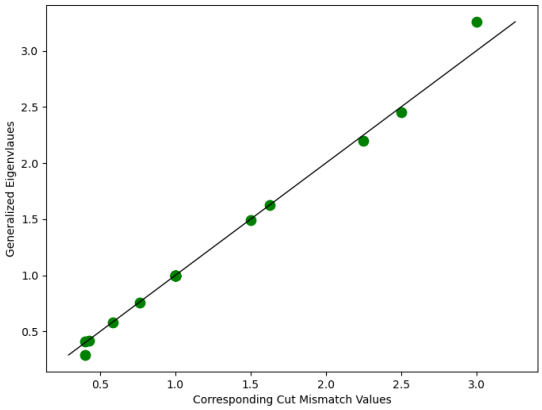

Figure 6: Generalized eigenvalues and their corresponding cut mismatches

Assuming lemma A.1 and A.2, for some $c, c' > 0$ sufficiently small, and by setting $\eta < c/\log n$ and $\sigma < c'\eta$, the algorithm ensures that the right sides of both equations 22 and 23 are at most $1/(12\eta^2)$. Then when $\pi^* = id$ (the identity permutation), these lemmas combine to imply:

$$\min_{i\in[n]} \widehat{X}_{ii} > \frac{1}{4\eta^2} > \frac{1}{6\eta^2} > \max_{i,j\in[n]:i\neq j} \widehat{X}_{ij} \tag{24}$$

with probability at least $1 - n^{-4}$. So, all diagonal entries of $\widehat{X}$ are larger than all off-diagonal entries, thereby achieving exact recovery (Fan et al., 2019).

A.9 Relation between Generalized Eigenvalues with Cut Mismatch

We selected two graphs from the MUTAG dataset and computed their generalized eigenvalues following the procedure for calculating the Generalized Graph Distance (GGD), which involves determining the node-to-node correspondence. Subsequently, we considerd all possible subsets of nodes and evaluate their corresponding cut mismatches. As shown in Figure 6, each generalized eigenvalue is closely associated with a cut mismatch. This empirical observation supports our hypothesis that the GGD between two input graphs is strongly correlated with structural mismatches in graphs.

A.10 Choosing ϵ for converting Laplacians to SPD matrices

Laplacian matrices are symmetric positive semi-definite (SPSD) matrices. To convert these to symmetric positive definite (SPD) matrices, we added a diagonal matrix with very small values (ϵ). We used 0.0001 as the small value (ϵ) in our experiments. When working with Laplacian matrices of a weighted or unweighted graph, values significantly smaller than the edge weights of that graph have a minimal effect on the transformation. We conducted additional experiments with different small values and included the results in Tables 4 and 5. In Table 4, we observed that in our specific case with the MUTAG graph dataset (Morris et al., 2020), where all graphs are unweighted, any value less than 0.001 has an almost negligible influence on the performance of the graph classification task. Additionally, when using values equal to or less than 0.001, the GGD value remains almost the same, as shown in Table 5.

A.11 Partial Node Features

Cutting-edge graph distance metrics like TMD rely on node attributes to compute the dissimilarity between graphs, resulting in more accurate outcomes when all attributes are available. However, acquiring datasets with complete node attributes is often unattainable in real-world scenarios, leading

Table 4: Classification accuracy using MUTAG dataset with different values of ϵ.

Value of ϵ	Classification accuracy	Value of ϵ	Classification accuracy
0.1	76.38 ± 7.89	1e-4	85.96 ± 5.26
5e-2	79.02 ± 6.58	1e-5	84.21 ± 5.26
1e-2	79.02 ± 5.26	1e-6	85.96 ± 5.26
5e-3	81.57 ± 7.89	1e-7	85.96 ± 7.89
1e-3	81.57 ± 7.89		

Table 5: GGD values using MUTAG dataset for different values of ϵ.

Value of ϵ	Normalized GGD of a random graph pair (MUTAG[85], MUTAG[103])	Average normalized GGD of 1000 pairs
0.1	0.712	0.727
5e-2	0.827	0.834
1e-2	0.952	0.959
5e-3	0.978	0.979
1e-3	0.996	0.995
1e-4	0.9996	0.9995
1e-5	0.99995	0.99996
1e-6	0.999996	0.999996
1e-7	1	1

to partially missing features In such scenarios when only partial node features are available, we compare TMD with GGD to better understand their differences. Table 6 shows that the TMD metric outperforms GGD at various levels when node features are fully accessible. However, when node features are randomly removed from the MUTAG dataset, the accuracy of TMD degrades substantially.

A.12 Approximate GGD on Small Graphs Using Extreme Eigenvalues

The largest and the smallest eigenvalues correspond to the most dominant mismatches in graph cuts and effective resistance distances, contributing the most to the total GGD value. Similarly, the second largest and smallest eigenvalues correspond to the next significant mismatched cuts. In our experiment, we obtain approximate GGD values using a few extreme eigenvalue pairs and compare them with the ground truth. Figure 7 illustrates the relative accuracy of the approximate GGDs, in which we observe that the top four pairs of extreme eigenvalues contribute 80% of the total GGD values. In addition, we conduct the SVC classification task and GNN correlation study using GGD with only 2 and 4 extreme eigenvalue pairs, respectively, and present the associated findings in Table 7.

A.13 Scalability of GGD Approximation on Very Large Graphs

For very large graphs, computing the full spectrum of eigenvalues becomes computationally expensive. To address this, we use approximate GGD strategy using only a small fraction of the extreme eigenvalues. Our results in Table 8 show that even with just 2% of the extreme eigenvalues, the approximation remains highly correlated with the original GGD value. Moreover, the percentage re-

Table 6: Comparison of correlation with GNN outputs and distance metrics with partial node features.

Dist Metric	0%	20%	50%	80%	100%
GGD	0.78	**0.78**	**0.77**	**0.77**	**0.77**
TMD, L = 3	**0.84**	**0.78**	0.72	0.63	0.61
TMD, L = 4	0.81	0.77	0.62	0.58	0.57
TMD, L = 5	0.80	0.75	0.65	0.58	0.53

Table 7: Performance of GGD using extreme eigenvalues only.

Task	Number of extreme eigenvalues		
	2	4	All
Correlation with GNN	0.74	0.76	**0.77**
Classification accuracy	81.50±6.85	83.87±7.56	**86.00±7.50**

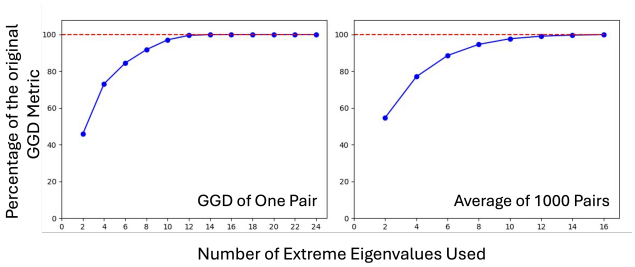

Figure 7: Percentage of the original GGD using numbers of extreme eigenvalues.

quired decreases with increasing graph size, significantly improving scalability. Table 9 demonstrates the resulting runtime improvements.

A.14 Comparison of Two Different Riemannian Metrics for SPD Matrices

The two most commonly used Riemannian metrics on the SPD manifold are the Affine Invariant Riemannian Metric (AIRM) and the Log-Euclidean Riemannian Metric (LERM) (Ilea et al., 2018; Thanwerdas & Pennec, 2023; Chen et al., 2024). AIRM is a Riemannian metric that remains invariant under affine transformations, meaning the metric is unaffected when matrices are transformed by any invertible operation. The geodesic distance between two SPD matrices, A and B, using AIRM is given by (You & Park, 2021; Moakher, 2005):

$$d_{\text{AIRM}}(A,B) = \|\log(A^{-1}B)\|_F = \left[\sum_{i=1}^{n} \log^2(\lambda_i(A^{-1}B))\right]^{1/2}. \tag{25}$$

On the other hand, LERM addresses some of the computational complexity challenges associated with AIRM by mapping SPD matrices to an Euclidean space through the matrix logarithm operation. In this Euclidean space, computations are simplified. The geodesic distance between two SPD matrices, A and B, using LERM is defined as (Huang et al., 2014):

$$d_{\text{LERM}}(A,B) = \|\log(A) - \log(B)\|_F. \tag{26}$$

In this paper, we primarily used AIRM to compute geodesics because of its stronger theoretical foundation and its ability to better explain graph cut mismatches. However, for comparison, we also conducted experiments using LERM. Figure 8 shows that the Graph Geodesic Distances computed using the LERM metric are highly correlated with those obtained using AIRM, though the GGD using AIRM demonstrates better performance overall. A detailed performance comparison of these two metrics is provided in Table 10.

Table 8: Correlation between GGD values and approximate GGD values using a portion of extreme eigenvalues for very large graphs, where n represents the number of nodes.

Extreme Eigenvalues Used	0.5%	1%	2%	4%	All
Correlation (n ∈ [5000, 5200])	0.8323	0.9361	0.9772	0.9902	1
Correlation (n ∈ [10000, 10500])	0.8666	0.9501	0.9808	0.9928	1
Correlation (n ∈ [15000, 16000])	0.8892	0.9599	0.9879	0.9952	1

Table 9: Runtime Comparison of Exact vs. Approximate GGD on Very Large Graphs

Node Numbers	Calculation Time		
	GGD	Aprx GGD (2% Eigs)	Aprx GGD (4% Eigs)
$n \in [5000, 5200]$	29.13s	7.11s	7.98s
$n \in [10000, 10500]$	241.33s	55.56s	56.29s
$n \in [15000, 16000]$	920.77s	162.30s	163.64s

Table 10: Comparison between Riemannian metrics for GGD calculation.

Riemannian metric	Correlation with GNN output	Classification accuracy
Affine-Invariant	**0.7786**	**86.00±7.50%**
Log-Euclidean	0.7634	84.38%

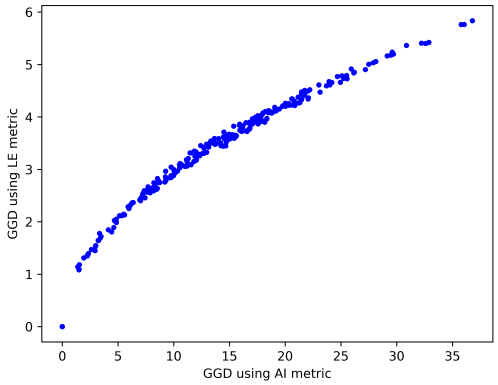

Figure 8: GGD between graph pairs using AI and LE Riemannian metric.

Table 11: Effect of epsilon in the calculation of GGD using normalized Laplacian matrices.

Value of ϵ	0.01	0.001	0.0001	0.00001
GGD using Laplacian matrices	16.235	16.775	16.832	16.838
GGD using normalized Laplacian matrices	384.097	254.440	188.345	165.332

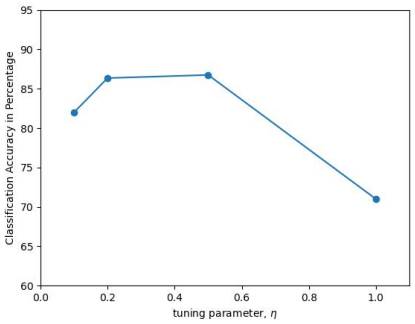

Figure 9: Classification accuracy vs GRAMPA tuning parameter.

A.15 Using Normalized Laplacians for GGD Calculation

In many existing studies, the normalized Laplacian matrix is commonly used to study spectral graph properties (Chung, 1997). The normalized Laplacian matrix of a graph G is defined as: $L_{norm} = I - A_{norm}$, where A_{norm} is the normalized adjacency matrix. The normalized adjacency matrix is expressed as (Chung, 1997):

$$A_{norm} = D^{-1/2} A D^{-1/2}, \tag{27}$$

where D represents the diagonal degree matrix, and A denotes the adjacency matrix of the graph.

Form equation 27, we can derive:

$$L_{norm} = I - D^{-1/2} A D^{-1/2} = D^{-1/2}(D - A)D^{-1/2} = D^{-1/2} L D^{-1/2}. \tag{28}$$

Similar to the Laplacian matrices of graphs, normalized graph Laplacian matrices are also symmetric and positive semi-definite. Therefore, it is necessary to add small values to the diagonal elements of these matrices. However, our experiments reveal that the GGD calculation is highly sensitive to this small value (ϵ), resulting in significant fluctuations across different values, as demonstrated in Table 11. Additionally, the geodesic distances computed with the modified normalized Laplacian matrices exhibit poor accuracy in both classification tasks and their correlation with GNN outputs.

A.16 Effect of tuning parameter η on Graph Matching

In the original work, it was suggested that the regularization parameter η needs to be chosen so that $\sigma \vee n^{-0.1} \lesssim \eta \lesssim 1/\log n$ (Fan et al., 2020). It is also mentioned that for practical cases, computing permutation matrix for different values of η in an iterative way can result in better accuracy. The GRAMPA uses $\eta = 0.2$ for all their experiments (Fan et al., 2020).
We used a few values of η in the classification problem using the MUTAG dataset and got that the best accuracy is obtained at $\eta = 0.5$. In Figure 9, the performance of the tuning parameter is demonstrated.

A.17 Sensitivity of GGD to Simple Graph Perturbations

Due to the simplicity and general applicability of the GGD, it can be readily computed between any pair of undirected graphs. To evaluate the robustness of GGD under simple structural changes, we

Table 12: Correlation between GGD values before and after applying different perturbation methods on smaller graphs (20–50 nodes)

Amount of Perturbation	1	2	3	4	5
Nodes Dropped	0.6934	0.6084	0.5896	0.5614	0.5368
Nodes Added	0.9976	0.9959	0.9943	0.9922	0.9900
Edges Removed	0.8375	0.7890	0.6644	0.6537	0.6332
Edges Added	0.9990	0.9982	0.9974	0.9964	0.9954

Table 13: Correlation between GGD values before and after applying different perturbation methods on larger graphs (200–500 nodes)

Amount of Perturbation	5	10	15	20	25	30
Nodes Dropped	0.8255	0.7734	0.6533	0.6116	0.5736	0.5563
Nodes Added	0.9991	0.9981	0.9963	0.9944	0.9915	0.9884
Edges Removed	0.9137	0.8857	0.8265	0.7989	0.7550	0.7226
Edges Added	0.9996	0.9992	0.9986	0.9979	0.9971	0.9963

conducted a series of experiments involving basic graph perturbations across both small and large graphs.

Our results show that node and edge additions typically have a minimal impact on the GGD values, suggesting that the metric is largely invariant to the inclusion of redundant elements. In contrast, node and edge drops exhibit noticeable effect. While a significant portion of these perturbations still yield GGD values comparable to the original graphs, some cases show some deviations. These deviations are associated with the removal of structurally important nodes or edges, which alters the graph topology. The quantitative results of these perturbation experiments are summarized Table 12, 13 and Figure 10, 11, 12, 13, 14, 15, 16, 17.

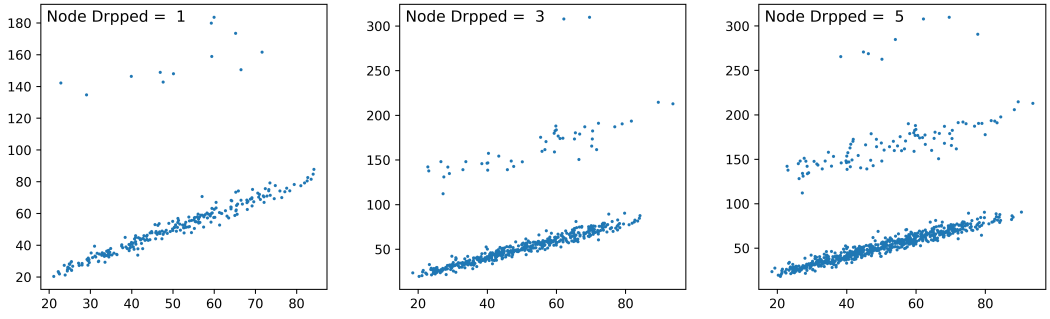

Figure 10: GGD (x-axis) vs GGD after **Node Drop** perturbation (y-axis) for smaller graphs.

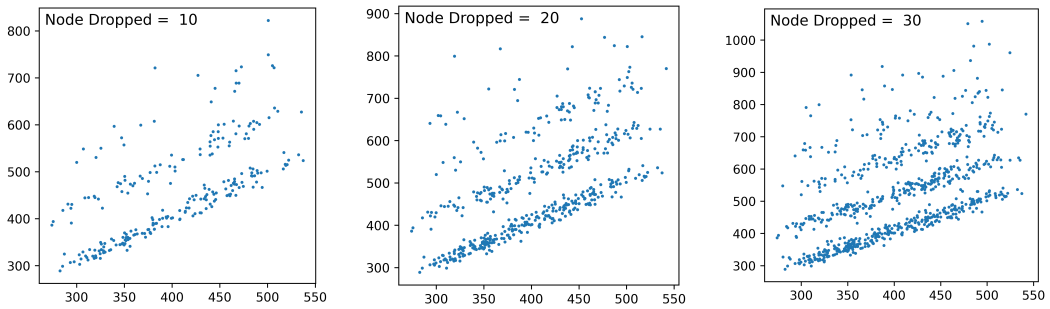

Figure 11: GGD (x-axis) vs GGD after **Node Drop** perturbation (y-axis) for larger graphs.

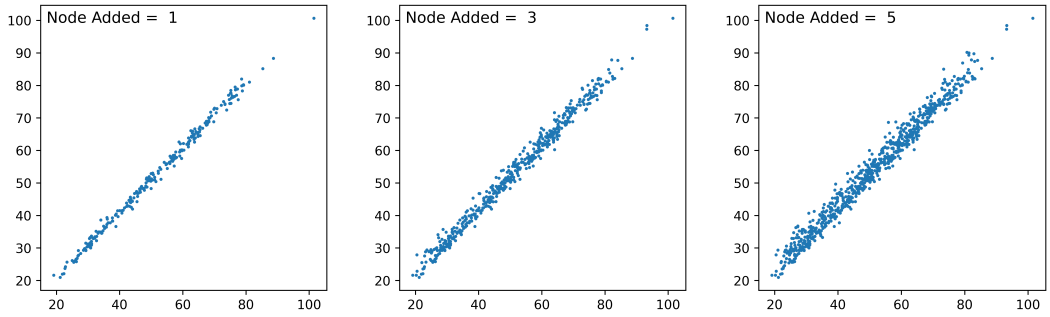

Figure 12: GGD (x-axis) vs GGD after **Node Addition** perturbation (y-axis) for smaller graphs.

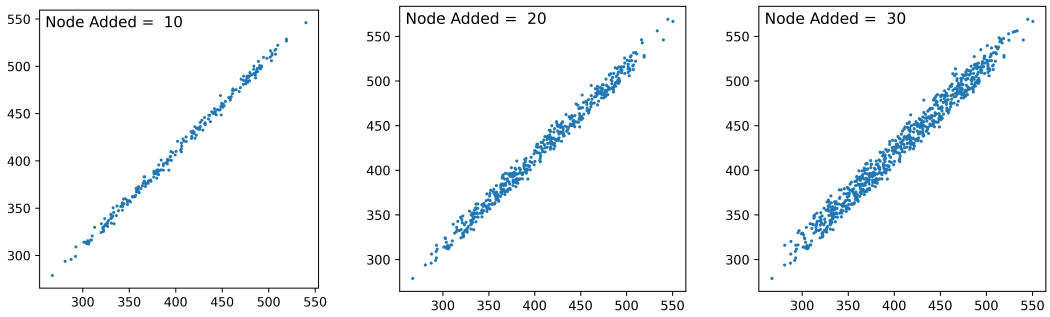

Figure 13: GGD (x-axis) vs GGD after **Node Addition** perturbation (y-axis) for larger graphs.

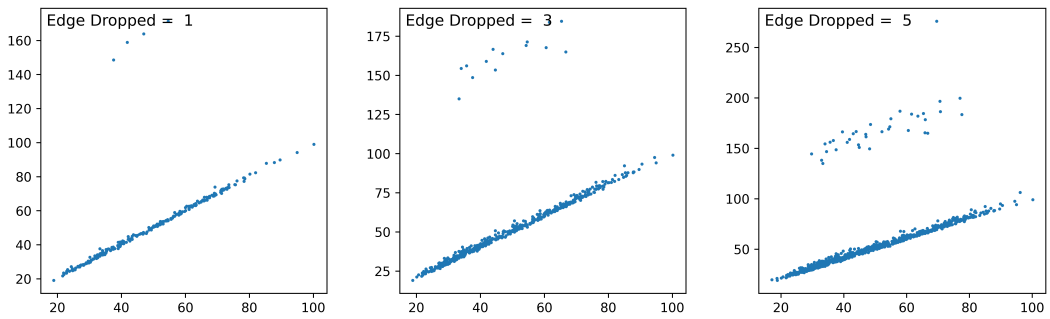

Figure 14: GGD (x-axis) vs GGD after **Edge Drop** perturbation (y-axis) for smaller graphs.

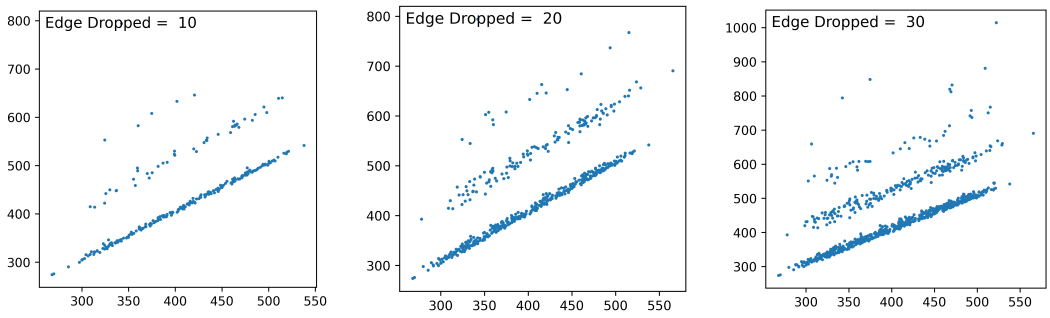

Figure 15: GGD (x-axis) vs GGD after **Edge Drop** perturbation (y-axis) for larger graphs.

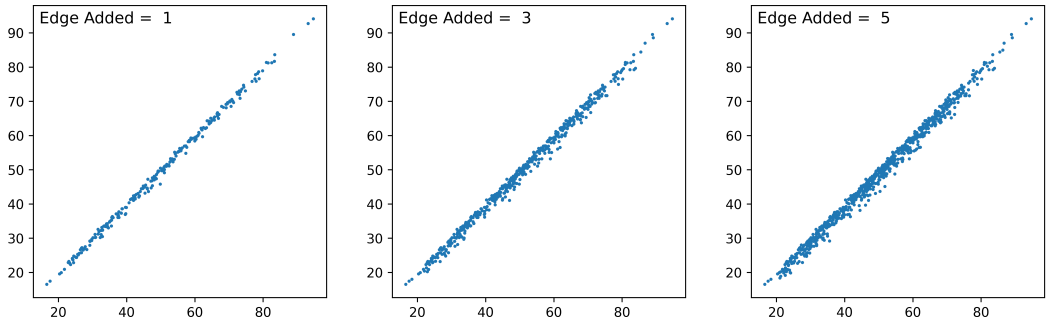

Figure 16: GGD (x-axis) vs GGD after **Edge Addition** perturbation (y-axis) for smaller graphs.

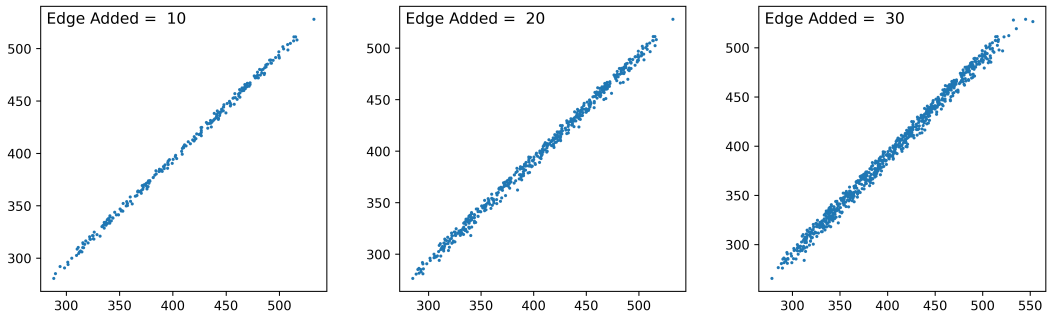

Figure 17: GGD (x-axis) vs GGD after **Edge Addition** perturbation (y-axis) for larger graphs.

A.18 Graph formation from Dataset using Probabilistic Graphical Models

We take similiar type of datasets such as MNIST, Fashion-MNIST, KMNIST, and USPS (Alvarez-Melis & Fusi, 2020) and convert them into connected graphs. To construct these graph structures, we use Probabilistic Graphical Models (PGMs), also known as Markov Random Fields (MRFs) (Cheng et al., 2024). PGMs are powerful tools in machine learning and statistical physics for representing complex systems with intricate dependency structures (Roy et al., 2009). They encode the conditional dependencies between random variables through an undirected graph structure. Recent studies have shown that the graph structure learned via PGMs can exhibit resistance distances that encode the Euclidean distances between their corresponding data samples (Feng, 2021).

We create a Feature Matrix (FM) U from the dataset, where each row represents a data sample, and the row U_p itself serves as the feature vector of that sample p, on this context- the pixel values. A dense k-nearest neighbor (k-NN) graph G_{dense} is initially constructed using the FM. To obtain the

final dataset graph $G_{dataset}$, spectral sparsification is applied by solving the convex optimization problem (Cheng et al., 2024):

$$\max_{\Theta} \quad F(\Theta) = \log\det(\Theta) - \frac{1}{k}\,\mathrm{Tr}\left(U^{\top}\Theta U\right) \tag{29}$$

where $\Theta = L + \frac{1}{\sigma^2} I$. Here, L is the graph Laplacian, $\mathrm{Tr}(\cdot)$ denotes the trace of a matrix, I is the identity matrix, and $\sigma^2 > 0$ represents the prior feature variance. To solve this the following lemma is used:

Lemma A.3. *Maximizing the objective function in Equation 29 can be achieved in nearly-linear time via the following edge pruning strategy equivalent to spectral sparsification of the initial dense nearest-neighbor graph. Specifically, edges with small distance ratios*

$$\rho_{p,q} = \frac{R_{\mathrm{eff}}(p,q)}{d_{\mathrm{euc}}(p,q)} = w_{p,q} \cdot R_{\mathrm{eff}}(p,q) \tag{30}$$

are pruned, where $R_{\mathrm{eff}}(p,q)$ denotes the effective resistance distance between nodes p and q, $d_{\mathrm{euc}}(p,q) = \|U_p - U_q\|_2^2$ represents the data distance between the feature of nodes p and q, and $w_{p,q} = \frac{1}{d_{\mathrm{euc}}(p,q)}$ is the weight of edge (p,q) (Cheng et al., 2024).

Computing the edge sampling probability $\rho_{p,q}$ for each edge (p,q) becomes computationally expensive for large graphs. To address this, an improved algorithm using a low-resistance-diameter (LRD) decomposition is proposed, extending the short-cycle decomposition (Chu et al., 2020) to weighted graphs. The method efficiently computes effective resistance to partition the graph into clusters, thereby enhancing the sparsification process. This results in a low-dimensional graph $G_{dataset}$ that retains important structural properties while reducing dimensionality (Cheng et al., 2024).

A.19 Limitations of GGD

Our current pipeline has three primary limitations. First, computationally, the end-to-end complexity is dominated by eigenvalue decomposition and the linear assignment step (both $O(n^3)$). Although this is comparable to or slightly better than some OT-based alternatives, scaling to very large graphs remains nontrivial. Appendix A.13 shows that using a small fraction of extreme eigenvalues preserves a high correlation with exact GGD and greatly reduces computation time, but this remains an approximation. Second, GGD is most reliable when comparing graphs of comparable size. When sizes differ dramatically, the size matching via graph coarsening introduces additional approximation error, though we expect such extreme mismatches to be rare in practical applications. Finally, like many graph distance metrics, our theory and experiments assume simple, undirected graphs. Extending GGD to directed, attributed, or higher-order graphs is a valuable direction for future work. We include these limitations to clarify the intended scope and reliability of GGD in practice and to guide when approximations or extensions may be required.

A.20 Experimental Setup

To evaluate the performance of the Graph Geodesic Distance (GGD) metric, we utilized graph datasets from the TUDataset collection (Morris et al., 2020). For small graphs, we used datasets like MUTAG and BZR, and for larger graphs, we selected PC-3H and SW-620H, which present more sizable networks. Detailed information about the datasets used is provided in Table 14.

Table 14: Brief description of graph datasets used.

Dataset name	Number of graphs	Average number of nodes	Average number of edges
MUTAG	188	17.93	19.79
PC-3H	27509	47.20	49.33
SW-620H	40532	26.06	28.09
BZR	405	35.75	38.36

While Classification tasks, each dataset was split into 90% training and 10% testing sets to ensure an unbiased evaluation process. When assessing the correlation with GNN, we trained a three-layer

GIN with 90% of all graphs from MUTAG and validated with the rest 10%. For the performance evaluation using graphs with partial node features, we took each dataset with node features and randomly removed a certain portion of features.

All experiments have been evaluated on a laptop with an Apple M1 chipset, featuring an eight-core CPU and a seven-core GPU.