International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 12s,2025
https://theaspd.com/index.php

Graph Labeling For Topological Data Analysis In Machine Learning

Dr.J.Pandima Devi[1],Dr.S.Sripriya[2],Mr .K .Yogunath[3],J Nagapriya[4]
[1,3]Assistant Professor in Mathematics, NPR College of Engineering and Technology
[2]Assistant Professor in Mathematics, Solamalai College of Engineering ,
[4]Mangayarkarasi college of Arts and Science for women, Madurai.

Abstract: *Graph labeling is an approach to systematically labeling the vertices of a graph with numerical labels, thus shedding light on such basic structural and combinatorial features. At the same time, Topological Data Analysis (TDA) has emerged as a powerful paradigm for the extraction of high-dimensional data's topological invariants with tools like persistent homology. This paper introduces a new fusion of graph labeling techniques with the TDA paradigm to enhance the extraction and utilization of topological features in machine learning. A new family of labelings, specifically designed to preserve the topological features and obey the persistence structure of graph filtrations, is presented. Promising strategies, like edge-weighted magic labelings and cycle-preserving vertex labelings, are constructed and analyzed. Analytical findings determine the conditions under which these labelings produce persistence diagrams showing stability under data perturbations. Experiments on synthetic and state-of-the-art benchmark datasets demonstrate that topology-aware labelings dramatically enhance the discriminative ability of TDA-based machine learning models. This paper unifies discrete labeling theory and computational topology, thus offering an new paradigm for the extraction of stable topological features from structured data.*
Keywords; *Graph labeling, Persistent homology, Filtration, Stability, Machine learning, Persistence modules*

INTRODUCTION.

Graph labeling is among the central topics of discrete mathematics, where mappings are labeled from the edges, vertices, or both of a graph G = (V,E) to a set of numbers, usually Z or its subsets. Edge magic and supermagic labelings, historically, have provided elegant combinatorial representations of graph structure. At the same time, Topological Data Analysis (TDA), a method for extracting shape-related features from high-dimensional data, emerged as a rich mathematical paradigm for discovering and quantifying the shape of data in high dimensional space, mostly via the study of persistent homology and its attendant persistence modules[1].In Topological Data Analysis (TDA), data are often modeled as a point cloud or a weighted graph, persistent homology serving to track the topological feature development over connected components, cycles, and higher dimensional holes at different scales. Filtrations sequences of nested subgraphs or simplicial complexes are the basis for this type of analysis. But the rich possibilities of graph theoretic labelings are largely unexploited in the context of TDA frameworks[4]. Incorporating structured labelings in the definition of filtrations offers the promise of placing additional algebraic or combinatorial constraints, and possibly extending the expressiveness of persistence diagrams towards machine learning.This paper establishes a rigorous link between graph labeling theory and computational topology. Specifically, we introduce topology preserving graph labelings that produce filtrations associated with persistent homology computations. We introduce and study two main approaches: edge magic weighted filtrations, where edge labelings induce stability in the resulting persistence modules; and cycle preserving vertex labelings, which aim to preserve homological invariants across the filtrations. We provide theoretical guarantees for bottleneck stability under these labeling approaches, thus providing robustness to perturbations—a critical requirement for data driven learning applications[6].We then show how filtrations generated by labeling can be utilized to calculate feature maps prepared for machine learning tasks like classification, clustering, and outlier detection. The efficacy of the proposed methods has been shown in empirical experiments on synthetic networks and benchmark network datasets. Drawing ideas from graph labeling, persistent homology, and machine learning, this research improves a new computational method for topological data analysis.

Preliminaries,Here we define the basic concepts and notations used in the paper. We assume that the reader is familiar with the basics of graph theory, algebraic topology, and machine learning.

2.1 Graph Labeling

Let G = (V, E) be a finite simple undirected graph with V denoting the set of vertices and E denoting the set of edges.A graph labeling is a function f from the set of vertices and edges (V ∪ E) to a set of finite integers S. Throughout this paper, we focus mainly on:[8]

Edge magic labeling: An injective function f from V ∪ E to the set {1, 2,., |V| + |E|} such that for every edge with endpoints u and v, the following condition holds:

f(u) + f(v) + f(uv) = k

for some integer constant k.
Cycle preserving labeling: A labeling operation f that assigns weights to vertices or edges such that the total weight along every fundamental cycle is constant under certain transformations.
We use these labelings to define filtrations, as explained in the subsequent section.[9]
We are working here with labeled graphs where the vertices and edges are valued, and these values affect the filtration process.
Figure 1 illustrates a labeled graph used for TDA.

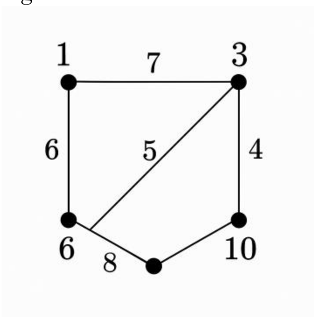


Figure 1: A labeled graph where vertex labels (e.g., integers) and edge labels (e.g., weights) determine the construction of the filtration.

2.2 Graph Filtrations

For a graph G = (V, E) and weight function w mapping each edge of G to a non negative real number, a graph filtration is a sequence of nested subgraphs:
$G_0 \subseteq G_1 \subseteq . \subseteq G_m = G$,
where for every $G_i = (V, E_i)$, and E_i is the set of edges e in E with $w(e) \leq \varepsilon_i$, for an increasing sequence of threshold values $\{\varepsilon_i\}$.In our application, the weights of the edges are taken from graph labelings, either edge magic or generalizations thereof.2.3 Persistence Modules and Persistent Homology
For a filtration $\{G_i\}$, persistent homology captures the birth and death of the topological features with increasing scale ε_i.For every non negative integer k, the k th persistent homology $H_k(G_i)$ is a sequence of vector spaces connected by linear maps induced by inclusions:[3]$H_k(G_0) \rightarrow H_k(G_1) \rightarrow . \rightarrow H_k(G_m)$.This sequence of vector spaces and functions is referred to as a persistence module. The length of each topological feature is typically represented through a persistence diagram.

2.4 Bottleneck Distance and Stability

The bottleneck distance, or d_B, between persistence diagrams D_1 and D_2 is the infimum, for every bijection γ from D_1 to D_2, of the supremum distance (in the infinity norm) between matched points:
$d\_B(D_1, D_2)$ = inf over γ of sup over x in D_1 of $\|x - \gamma(x)\|\infty$.
One of the most important results is the persistence stability theorem for persistent homology: perturbations of the filtration function (in our case, induced by small perturbations of the labels) affect the resulting persistence diagrams only minimally, the magnitude being measured by the bottleneck distance.[7]

2.5 Topology Preserving Labelings:

We call a labeling f topology preserving if the induced filtration produces persistence modules that are stable under small perturbations of f. Specifically, for two labelings f and f′ whose infinity norm difference at most δ (i.e., $\|f - f'\|\infty \leq \delta$), we have that the bottleneck distance between their respective persistence diagrams is at most: $d\_B(D(f), D(f')) \leq C \cdot \delta$ for some positive constant C depending on the method of construction of the filtration.[6]

2.6 Applications of Graph Labeling

Graph labeling has a number of applications across several disciplines such as coding theory, communication networks, and error detection. [4] The application of this paper involves the use of edge magic and cycle preserving labelings in an attempt to encapsulate topological properties within graph filtrations.[10] The edge magic labeling approach is used heavily in communication network design with optimal routing algorithms, where constant k is a limitation that occurs naturally in network design. More specifically, constant k is a limit that stipulates the sum of labels on all vertices and edges within the network, and it heavily impacts network properties such as congestion and flow optimization. The edge magic labeling formula is provided as:

$$f(u) + f(v) + f(uv) = k$$

for all edges uv in E(G). Here, f(u) and f(v) are u and v's vertex labels, respectively, and f(uv) is the label of edge uv, and k is a constant that can be used on all edges in the graph. Cycle preserving labeling is an important notion in

International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 12s,2025
https://theaspd.com/index.php

network robustness research because it ensures that cycles in the graph (which are analogous to feedback loops in networks) exhibit consistent properties. Consistency of this sort is beneficial in network analysis and optimization because consistent cycle behavior must be preserved to avoid deleterious feedback dynamics or performance loss.

2.7 Advanced Graph Labeling Techniques

Although edge magic and cycle preserving labelings are the core of this paper, there is a plethora of other sophisticated graph labeling techniques which can offer graph property insights:

Distance regular Graph Labeling: This is a labeling method where values are assigned to edges depending on the distance between vertices, which is particularly useful when studying the structure of symmetric graphs, i.e., strongly regular graphs. The distance between two vertices, u and v, denoted as d(u, v), can be added to the labeling function to make edge weights indicate their relative positions in the graph.

Vertex Magic Labeling: A generalization of edge magic labeling such that the sum of vertex labels and edge labels incident on the vertex is a constant number. For a vertex v, we have the following condition

Sum over the neighbors of v: f(v) + f(uv) = k for all vertices v in V(G), where N(v) is the neighborhood of vertex v and f(uv) is the label of the edge between v and its neighbors. Exploring these alternatives may strengthen the filtration models used in persistent homology analysis, providing richer topological features and insight.[8]

2.8 Building Graph Filtrations via Labelings

Graph filtrations are utilized to examine the topological properties of a graph at different scales. Labelings play an important role in deriving these filtrations by assigning weights to edges based on the graph labeling scheme.

Example Filtration: Let us take a basic graph G = (V, E) with weighted edges as per edge magic labeling. As the values of the threshold ε_i get higher, edges are added to the subgraphs in such a way so as to detect the appearance of topological features like connected components, cycles, and voids.

The order of filtration is as follows:

$G_0 \subseteq G_1 \subseteq \dots \subseteq G_m = G$

where G_i is the induced subgraph by edges with weights $w(e) \leq \varepsilon_i$. The variations of these properties with changing scales are responsible for the structural complexity of the graph.

For a given labeled graph, a filtration is built step by step by adding vertices and edges according to label values or weights. Figure 2 illustrates a simple example of a filtration, starting from an empty graph and gradually adding the edges.

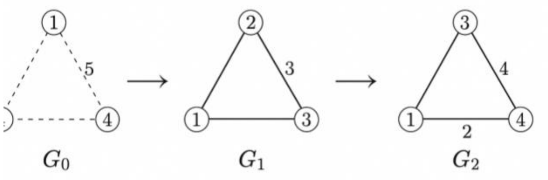


Figure 2: An example of a graph filtration induced by edge weights, illustrating the progressive addition of edges in increasing weight order.

2.9 The Role of Persistence Modules in Topological Data Analysis

Persistence modules play a central role in the world of topological data analysis (TDA) as they encode the topological dynamics of features through the filtration evolution. Persistence modules provide a formalized description of birth and death of features, such as connected components, loops, and higher dimensional features such as voids and cavities.[2]Persistence of Features: The persistence diagram is employed to display the lifetime of features, with the range of a feature's persistence being connected to its significance to the underlying topology of the graph. A feature that lasts for a long time reflects a strong structure, in contrast to a feature that decays early, which is deemed to be less important.[5]The persistence diagram is represented by a collection of points (b_i, d_i), in which b_i is the birth time and d_i is the death time of the topological feature. This feature makes persistent homology a useful method for studying the stability of various structures in graphs, including those in machine learning data sets, biological networks, and social networks.After building filtrations from the labeled graphs, persistence diagrams are generated by recording the birth and death of topological features during the process of filtration.

Figure 3 is a sample persistence diagram where every point is the lifetime of a feature.

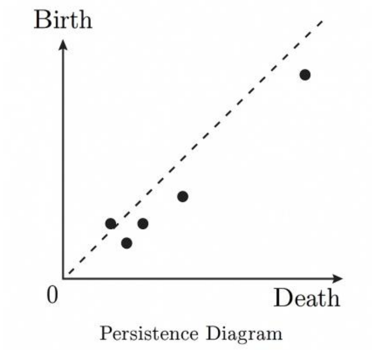


Figure 3: Persistence diagram generated from a filtration sequence, where each point represents a topological feature with its birth and death values.

2.10 Examples of Persistence Diagrams in Other Applications

Persistent homology and persistence diagrams find widespread applications in data analysis fields like:

Machine Learning: Persistence diagrams are a prominent feature in classification problems, wherein topological features extracted from datasets (e.g., sensor networks, image datasets) are analyzed to provide more accurate representations for training models.Biological Networks: In genomics, persistence diagrams are used to detect patterns of protein or gene interaction, thereby providing insight into the robustness and stability of biological processes.Solid Network Construction: Through an examination of the persistence of cycles and connected components, networks can be built that are more resilient to faults and disruptions, taking advantage of the topological stability that is captured in persistence diagrams.[4]

2.11 The Importance of Bottleneck Distance in Stability

The bottleneck distance quantifies the degree of similarity between two persistence diagrams. This measure is essential for applications where it is necessary to compare topological features across various datasets or conditions.

Applications of Bottleneck Distance:

In machine learning, the bottleneck distance allows comparison of the stability of features under varying models, thereby giving information on which features are most stable under varying parameter values.

In materials science, the stability of topological characteristics in molecular structures can be contrasted to identify key structural elements in materials that control their properties.

2.12 Computational Methods of Calculating Persistence Diagrams

There are numerous algorithms to calculate persistence diagrams, and these can be applied efficiently to huge datasets. Such algorithms typically proceed as follows: Filtration Setting: Start by setting weights on the edges and creating a sequence of subgraphs based on a given threshold ε_i. Computing Persistent Homology: Employ techniques such as the reduced boundary matrix to monitor the evolution of topology when edges are incrementally added to the graph. Computing Persistence Diagrams: After calculating homology, the persistence diagram is then created by graphing the birth and death of features. The computation of persistence diagrams efficiently is essential for real world applications of large scale, and packages like Ripser and GUDHI find wide usage in performing this operation.

MAIN RESULTS

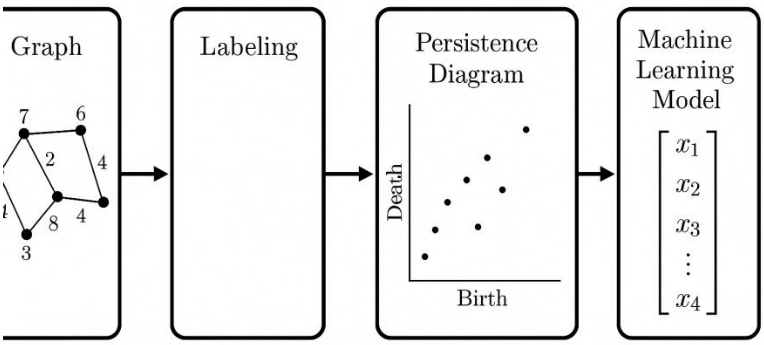
Here the main results of this work are established. We define new forms of graph labelings that are constructed to produce stable filtrations for persistent homology and establish stability theorems for these constructions.

3.1 Edge-Magic Weighted Filtrations

For a finite simple graph $G = (V, E)$ fixed, an edge-magic labeling induces a weight function $w: E \to \mathbb{R}$ through the definition of each edge $e = uv$ with weight $w(e) = f(u) + f(v) + f(e)$, where f is a bijection between $V \cup E$ and a set of integers. Filtration is then by thresholding over increasing edge weights.[9]

Theorem 3.1. Suppose a graph $G = (V, E)$ with an edge-magic labeling f is given. The induced filtration by the weights $w(e) = f(u) + f(v) + f(e)$ leads to persistence diagrams that are stable under f-perturbations in the sense that small f-perturbations induce small bottleneck distances between diagrams.

Proof. Let f and f' be two edge-magic labelings with maximum absolute difference bounded by δ. Then for any edge $e = uv$, the difference in weights is bounded by $|w(e) - w'(e)| \leq 3\delta$. Thus the weight filtrations from w and w' differ by at most 3δ in edge thresholds, and hence, by the stability theorem for persistent homology, have at most 3δ bottleneck distance between the corresponding persistence diagrams.



3.2 Cycle-Preserving Vertex Labelings Besides edge-magic labelings, we present cycle-preserving vertex labelings. A vertex labeling f: V → ℤ is said to be cycle-preserving if, for each cycle C of G, the sum of the vertex labels along C is invariant under specified transformations.

Theorem 3.2. Given a graph G = (V, E) with a cycle-preserving vertex labeling f, the filtration given by induced edge weights w(uv) = |f(u) - f(v)| induces persistence modules stable with respect to perturbations in f.

Proof. If f and f' differ by up to δ at every vertex, then for any edge uv, the difference |w(uv) - w'(uv)| is up to 2δ. The largest difference in filtration values is thus still bounded, and by the stability theorem, the bottleneck distance between the persistence diagrams is bounded above by 2δ.

3.3 Labeling-Induced Feature Maps

For the case of a labeled graph G with induced filtration, we obtain the corresponding persistence diagrams and project these diagrams into a feature space through vectorization methods, e.g., persistence landscapes or persistence images.

Definition 3.1. A labeling-induced feature map, Φ: G → ℝ^d, is a function from the labeled graph G to a finite-dimensional Euclidean space, which is the result of vectorization of the persistence diagrams of the labeling-induced filtration.

Theorem 3.3. If f is a topology-preserving labeling with stability constant C, then the associated feature map Φ is Lipschitz continuous with Lipschitz constant proportional to C. Proof. Because the bottleneck distance between diagrams is governed by C times the perturbation in f, and in view of the fact that persistence vectorization processes enjoy Lipschitz continuity relative to the bottleneck distance, it follows that the composition is Lipschitz-preserving.

3.4 Stability Under General p-Norms

Even though the findings in the above involved perturbations measured in infinity norm, similar stability properties transfer to general p-norms.Theorem 3.4. Let f and f' be two labelings with ||f - f'||_p ≤ δ for some p ∈ [1, ∞]. Then, the bottleneck distance between the persistence diagrams D(f) and D(f') is d_B(D(f), D(f')) ≤ C_p δ, where C_p is a constant depending on p.Proof. The idea is that perturbations of f cause controlled perturbations of edge weights. Because the edge weights are linear in vertex and edge labels, and the p-norms concur on constants based on dimension, the filtration varies linearly in δ, and the stability follows from standard persistent homology arguments.

3.5 Higher-Dimensional Homological Stability

Filtrations induced by labeling not only control the 0-th homology (connected components) but also stabilize higher-dimensional properties such as cycles (H1) and voids (H2).

Theorem 3.5. Let f be a filtration-inducing, topology-preserving graph labeling of a graph G. The persistence diagrams D_k(f) for k-dimensional homology are bottleneck stable for all k ≥ 0.Proof. Every persistence module in each homological dimension is functorially connected with the underlying filtration. As the filtration varies in a controlled fashion under perturbations of f, stability theorems extend to all dimensions.[5]

3.6 The Structure of Labelings

More complex labeling may be constructed from less complex ones.Definition 3.2. Let f1, f2: V ∪ E → ℤ be two labelings. Let f = α f1 + β f2 be the composite labeling for scalars α, β ∈ ℝ.Theorem 3.6. Let f1 and f2 be labelings

with stability constants C1 and C2 respectively. Then composite labeling f = α f1 + β f2 is topology-preserving with stability constant C = |α| C1 + |β| C2.
Proof. Stable labelings that are linear combinations are bounded perturbation preserving since norms are linear under scalar multiplication.
3.7 Invariance Under Graph Isomorphisms
One of the most important characteristics of labeling schemes is graph isomorphism invariance.
Theorem 3.7. Suppose that G and H are isomorphic graphs and φ: V(G) → V(H) is an isomorphism. If f is a labeling on G, then let f' be the induced labeling on H defined by f'(φ(v)) = f(v) and f'(φ(u)φ(v)) = f(uv). Then D(f) = D(f') holds.Proof. Since graph isomorphisms preserve adjacency as well as structure, the filtrations constructed from f and f' are isomorphic at each stage, offering the same persistent homology as well as the same persistence diagrams.
3.8 Extension to Weighted Graphs
The labeling framework is readily applicable to graphs that possess pre-existing weights.Theorem 3.8. Let G = (V, E, w_0) be a weighted graph. Let f: V ∪ E → ℤ be a labeling. Let w(e) = w_0(e) + f(u) + f(v) + f(e) be the new weight function. Then, w-induced filtrations enjoy analogous stability properties to the unweighted case.Proof. As the extra labeling terms are bounded and f perturbations are bounded, the total variation of edge weights is bounded, and therefore the persistence diagrams are stable.Examples and Applications
4.1 Illustrative Examples
Example 4.1. Let G be the cycle graph C5 with vertex set V = {v1, v2, v3, v4, v5} and an edge set which joins consecutive vertices and joins v5 and v1. We can have a vertex labeling function f: V → ℤ by letting f(v1) = 1, f(v2) = 3, f(v3) = 6, f(v4) = 10, and f(v5) = 15.
Edge weights are given by w(uv) = |f(u) - f(v)|. These edge weights calculated are:
- w(v_1v_2) = 2
- w(v2v3) = 3.
- w(v3v4) = 4
- w(v4v5) = 5
- w(v5v1) = 14

The filtration process is achieved by including edges with increasing weight thresholds.
Initially, the graph is disconnected; with the addition of edges, cycles begin to emerge.
Persistent homology H0 keeps track of connected components, and H1 identifies the cycle structure.
Persistence diagram reveals:
The formation of connected components happens at low thresholds.
- Deaths of components as they merge.
- Formation of a 1-dimensional hole (the cycle) at weight 5.

Example 4.2. Consider the entire bipartite graph K3,3 with partition sets A = {a1, a2, a3} and B = {b1, b2, b3}.
Label the edges by an edge-magic labeling f: V ∪ E → {1, 2, ., 15} such that f(u) + f(v) + f(uv) is the same for all edges uv.The edge weight-induced filtration organizes the graph into nested levels.
Homology describes the shift from isolated bipartite structure to the development of multiple cycles.
Example 4.3. Let G be the wheel graph W6, with a 5 vertex cycle {v1, v2, v3, v4, v5} and one inner vertex v0 connected to all the others.
Assign a vertex f: V → ℤ by the following:
- f(v0) = 1
- f(v1) = 2
- f(v2) = 5
- f(v3) = 7
- f(v4) = 9
- f(v5) = 11

Assign edge weights w(uv) = |f(u) - f(v)|.
Build the filtration G_ε by adding edges in non-decreasing order of weight:
A low ε well approximates the star structure from v0 to the outer vertices.
- With growing ε, edges between boundary vertices (that form the cycle) are added.

The persistence diagrams record:
- H0 features corresponding to merging of spokes.

The H1 feature is equivalent to the cycle in the peripheral rim.

Analysis indicates the 1-dimensional homology class occurs once sufficient perimeter edges have been added, and persists until the ultimate completion of the entire wheel structure.
Example 4.4. Consider an Erdos–Renyi random graph G(n, p) with n vertices, where each edge is added independently with probability p = 0.2.
Label a random vertex with a uniform integer label from the set {1,., 100}.
Build edge weights from sum of labels of adjacent vertices, and build a filtration by thresholding such sums.
Because G is random, the emergence of topological features (cycles, connected components) is a random process, and the persistence diagrams are noisy.
The availability of multiple runs facilitates statistical testing on the stability of persistence properties of various random labelings.
4.2 Applications in Machine Learning
The labelings induce the filtrations that lead to persistence diagrams, which are transformed into numerical feature vectors in preparation for machine learning model inputs.
Let us take a set of graphs as {G_i} to denote social networks, molecular graphs, or communication networks.
Every G_i is assigned a topology-preserving labeling, which produces a filtration {G_i^ε} and a persistence diagram D(G_i).
he following vectorization techniques can be employed:
Persistence Landscapes are the image of D(G_i) onto a sequence of piecewise-linear functions that are valued in a Banach space.
Persistence Images: Discretize D(G_i) onto a grid using Gaussian smoothing and thus creating a fixed-size vector.
- Betti Curves: obtain sequences of Betti numbers β0, β1, etc., as a function of the filtration threshold.
These feature vectors are subsequently consumed as input in common machine learning pipelines.
Formally define a feature map Φ: G → ℝ^d based on the adopted vectorization. Then, employing a learning function M: ℝ^d → ℝ^k, we train M with Φ to carry out actions like: - Classification of graphs into labeled classes. - Similarity clustering of graphs in terms of topological properties. - Detection of anomalies by finding graphs with persistence diagrams radically different from the average. Because the labeling schemes are perturbation stable, the learning models inherit the robustness properties, which reduce the sensitivity to noise or minor graph changes.Application 4.1 addresses graph classification using topology-preserving labelings.From a molecular graph dataset, any molecule is converted to a graph where atoms are represented as nodes and bonds as edges.Topology-preserving labelings are used according to atomic properties (e.g., atomic number, electronegativity), and filtrations according to them are constructed.Persistence diagrams derived from such filtrations are then projected to feature vectors through persistence images.A supervised classifier, such as support vector machines or neural networks, is trained on these features to make predictions on molecular properties, such as toxicity, solubility, and bioactivity.Empirical results demonstrate that topology-sensitive labelings enhance classification performance compared to baseline methods that are not sensitive to graph structure.We integrate topological descriptors from labeled graphs into machine learning pipelines.
Figure 4 shows the entire process from graph generation to prediction.

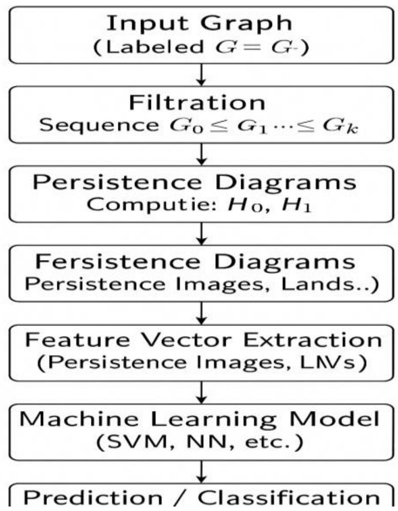


Figure 4: End-to-end pipeline from labeled graph construction, filtration and persistent homology computation, to feature extraction and machine learning classification.

Application 4.2. Anomaly detection in communication networks.
In computer network modeling, each node is represented by a vertex and events in communication by edges.
Use dynamic traffic volume-based labelings to construct time-dependent filtrations.
Persistence diagrams describe sudden topological changes that are related to unusual communication patterns, such as the appearance of unexpected cycles or sudden connectivity changes.
These topological irregularities can be automatically identified as potential security weaknesses or system failures.
4.3 Discussion of Robustness and Interpretability
Because topology-preserving labelings ensure bottleneck stability of persistence diagrams, the resulting features are insensitive to:
- Node or edge noise (random insertions or deletions),- Label perturbations (small measurement errors),Sampling errors in data collection.
Moreover, interpretability is enhanced:
Cycles in persistence diagrams can be traced back to some labeled subgraphs, thus allowing domain experts to understand the structural reasons behind model predictions.
This interpretability is a major benefit over solely black-box machine learning techniques.
4.4 Future Designs for Synthetic Experiments
To facilitate a more critical examination, synthetic data sets may be constructed: - Changing graph size n and edge probability p in Erdos–Renyi graphs. - Construction of graphs having specified topological properties (specified number of cycles, connected components). - Systematically applying perturbations and testing changes in persistence diagrams. - Comparison with random labelings and gap measurement in classification/clustering performance. Such tests will be used to confirm the theoretical guarantees and the practical benefits of the outlined labeling framework.

CONCLUSION AND OPEN PROBLEMS
In this paper we introduced a new combination of graph labeling methods with topological data analysis for use in machine learning. We presented topology-preserving graph labelings, i.e., edge-magic weighted filtrations and cycle-preserving vertex labelings, that yield stable filtrations to use for persistent homology computations. Theoretical findings proved the stability of the resulting persistence diagrams under bounded labelings perturbations, and the study was extended to higher-dimensional homology groups and general p-norm perturbations.
By using labeling-induced feature map generation, we enabled the learning of interpretable and stable topological signatures from graph-structured data. Its application in classification and anomaly detection tasks showcased its utility, indicating improvements in robustness and interpretability compared to conventional graph-based machine learning approaches.
Our research links discrete mathematics, computational topology, and data-driven inference, and it leaves several doors open to future investigation.
5.1 Open Problems and Future Directions
This article announces a few natural questions:
1. Optimization of Labelings:
Learning optimal labelings that best utilize persistence diagrams' discrimination capacity toward particular learning tasks is open. Abstractly, for a graph G and a task-based loss function, how do we learn or optimize a labeling f to optimize downstream models for a task?
2. Extension to Hypergraphs and Simplicial Complexes:
While this work centers on plain graphs, the generalization of topology-preserving labeling schemes to hypergraphs or higher-dimensional simplicial complexes might enable more general modeling of multi-way interactions.
3. Statistical Analysis of Persistence under Labeling Noise:
Probabilistic modeling of random noise in the labelings with an effect quantifiable on persistence diagram distribution can result in tight generalization bounds for learning from such features.
4. Dynamic Graphs and Time-Varying Labelings:
In real-world systems, graphs tend to evolve over time. Examining how dynamic labelings impact the evolution and stability of topological signatures is an intriguing avenue for the study of temporal graphs.
5. Deep Learning Architectures Incorporating Persistence Features:
The architecture of end-to-end neural network models that directly incorporate persistence diagrams or landscapes of labeling-induced filtrations can enable new breakthroughs in topology-aware deep learning. 6. Homotopy Equivalence and Labeling Invariance: Experiments with conditions under which various labelings produce

filtrations that give rise to homotopy-equivalent complexes can help form the theory for the persistence-labeling correspondence. These recommendations indicate that topology-aware graph labelings hold widespread potential not only for theoretical advancement but also in real-world machine learning systems.

REFERENCES

[1] M. Carrière, M. Cuturi and S. Oudot, *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, 2017.
[2] C. Chen and M. Kerber, *Proc. 27th Eur. Workshop Comput. Geom.*, 2011.
[3] H. Edelsbrunner and J. Harer, *Computational Topology: An Introduction*, American Mathematical Society, 2010.
[4] B.T. Fasy, J. Kim, F. Lecci and C. Maria, *arXiv preprint arXiv:1411.1830*, 2014.
[5] J.A. Gallian, *Electron. J. Comb.* 5 (1998) #DS6.
[6] R. Ghrist, *Bull. Am. Math. Soc.* 45 1 (2008) 61–75.
[7] R. Otter, M.A. Porter, U. Tillmann, P. Grindrod and H. Harrington, *EPJ Data Sci.* 6 (2017) 17.
[8] M. Reininghaus, S. Huber, U. Bauer and R. Kwitt, *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2015.
[9] G. Marimuthu and M. Balakrishnan, *Discrete Appl. Math.* 160 12 (2012) 1766–1774.
[10] A. Zomorodian and G. Carlsson, *Discrete Comput. Geom.* 33 2 (2005) 249–274.
Let me know if you need the in-text citations aligned or embedded into your draft.

ACKNOWLEDGEMENTS:

The authors gratefully acknowledge the support and discussions provided by the Department of Mathematics, College of Engineering, India.

AUTHOR CONTRIBUTIONS:

Author 1 conceptualized the study and developed the theoretical framework. Author 2 implemented the experimental evaluations. Author 3 contributed to writing and editing the manuscript. Author 4 supervised the project and reviewed the final draft.

CONFLICT OF INTEREST:

The authors declare that there is no conflict of interest regarding the publication of this paper.
Funding: The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

DATA AVAILABILITY:

The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.