Machine Learning (2024) 113:7267–7292
https://doi.org/10.1007/s10994-024-06616-w

Persistent Laplacian-enhanced algorithm for scarcely labeled data classification

Gokul Bhusal[1] · **Ekaterina Merkurjev**[1,2] · **Guo-Wei Wei**[1,3,4]

Received: 31 May 2023 / Revised: 12 August 2024 / Accepted: 20 August 2024 /
Published online: 13 September 2024

Abstract

The success of many machine learning (ML) methods depends crucially on having large amounts of labeled data. However, obtaining enough labeled data can be expensive, time-consuming, and subject to ethical constraints for many applications. One approach that has shown tremendous value in addressing this challenge is semi-supervised learning (SSL); this technique utilizes both labeled and unlabeled data during training, often with much less labeled data than unlabeled data, which is often relatively easy and inexpensive to obtain. In fact, SSL methods are particularly useful in applications where the cost of labeling data is especially expensive, such as medical analysis, natural language processing, or speech recognition. A subset of SSL methods that have achieved great success in various domains involves algorithms that integrate graph-based techniques. These procedures are popular due to the vast amount of information provided by the graphical framework. In this work, we propose an algebraic topology-based semi-supervised method called persistent Laplacian-enhanced graph MBO by integrating persistent spectral graph theory with the classical Merriman–Bence–Osher (MBO) scheme. Specifically, we use a filtration procedure to generate a sequence of chain complexes and associated families of simplicial complexes, from which we construct a family of persistent Laplacians. Overall, it is a very efficient procedure that requires much less labeled data to perform well compared to many ML techniques, and it can be adapted for both small and large datasets. We evaluate the performance of our method on classifica-

Editor: Hendrik Blockeel.

✉ Ekaterina Merkurjev
merkurje@msu.edu

Gokul Bhusal
bhusalgo@msu.edu

Guo-Wei Wei
Weig@msu.edu

[1] Department of Mathematics, Michigan State University, East Lansing, MI 48824, USA

[2] Department of Computational Mathematics, Science and Engineering, Michigan State University, East Lansing, MI 48824, USA

[3] Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48824, USA

[4] Department of Biochemistry and Molecular Biology, Michigan State University, East Lansing, MI 48824, USA

tion, and the results indicate that the technique outperforms other existing semi-supervised algorithms.

Keywords Topology-based framework · Graph MBO technique · Persistent Laplacian · Scarcely labeled data

1 Introduction

Machine learning has had tremendous success in science, engineering, and many other fields. However, most machine learning algorithms require large amounts of labeled data in order to build a good model and make accurate predictions. At the same time, obtaining sufficient amounts of labeled data can be very challenging for many applications as it is often expensive and time-consuming, and sometimes requires experts in the field. On the other hand, unlabeled data is often available in abundance. Therefore, semi-supervised learning (SSL), the main recent approaches of which have been outlined in a survey (Van Engelen and Hoos, 2020) and which utilizes mostly unlabeled data and much less labeled data for training, has garnered significant attention in the machine learning community. In particular, one class of semi-supervised learning algorithms that has gained popularity is graph-based semi-supervised learning. The key goal of such methods is to use a graph structure, which is often similarity-based, and both labeled and unlabeled data points, for machine learning tasks. Here, a graph is often constructed with nodes and edges, where the nodes represent the labeled and unlabeled data set elements, and the edges contain weights that encode the similarity between pairs of data elements.

Overall, graph-based methods, such as those detailed in Sect. 2.1, have shown great promise for several reasons. First, a similarity graph-based framework provides valuable information about the extent of similarity between data elements through a weighted similarity graph, which is crucial for applications such as data classification. Additionally, graph-based methods yield information about the overall structure of the data. Second, graph-based methods are capable of incorporating diverse types of data, such as social networks, sensor networks, and biological interaction networks, using a graph structure. This flexibility makes them some of the most competitive methods across a wide range of applications. In addition, most real-world datasets exist in high-dimensional Euclidean space, but embedding the features into a graphical setting reduces the dimensionality of the problem.

Moreover, topological data analysis (TDA) has recently emerged as a powerful tool for analyzing complex data. The central technique of TDA is persistent homology (PH), which combines classical homology and geometric filtration to capture topological changes at different scales. However, PH has some limitations. Specifically, PH fails to capture the homotopic shape evolution of data during filtration. To overcome this limitation, Wang et al. (2020) introduces the concept of persistent spectral graphs, also known as persistent combinatorial Laplacians or persistent Laplacians (PLs). In particular, PLs are an extension of the standard combinatorial Laplacian to the filtration setting. It turns out that the harmonic spectra of the PLs return all topological invariants, and the non-harmonic spectra of the PLs provide information about the homotopic shape evolution of the data during filtration.

Motivated by the success of graph-based methods, persistent spectral graph theory, and semi-supervised learning, we propose a novel graph-based algorithm for data classification with low label rates by integrating similarity graph-based threshold dynamics with a family of persistent Laplacians and semi-supervised techniques. The proposed method, called

persistent Laplacian-enhanced graph MBO (PL-MBO), adapts the classical MBO scheme developed in Merriman et al. (1994) to a label-propagation-based graph framework for data classification. We validate our proposed algorithm using five benchmark data classification datasets.

Specifically, the motivation for the proposed method stems especially from the benefits of using the combination of graph-based techniques, semi-supervised procedures and persistent spectral graph theory to derive the proposed algorithm, each of which are equipped with their own important advantages.

For example, semi-supervised techniques significantly reduce the amount of labeled data needed for accurate predictions due to their use of the information from the (vastly available) unlabeled data. This is crucial since labeled data is scarce for many applications. Since most machine learning approaches rely on large labeled sets to perform well, using semi-supervised techniques provides our method with the advantage of good accuracy even in the common scenario of low label rates.

In addition, integrating persistent spectral graph theory is beneficial since the theory improves upon the persistent homology, the main workhorse of topological data analysis. In fact, its kernel produces the same topological information as persistent homology, but its non-harmonic spectrum offers additional shape evolution of the data, thus providing the algorithm with more information which can often improve accuracy. This demonstrated via ablation studies in Table 4.

Lastly, using (similarity) graph-based techniques offers several aforementioned advantages such as providing valuable information about the extent of similarity between (labeled or unlabeled) data elements through a weighted similarity graph, which is crucial for applications such as data classification. Moreover, among semi-supervised learning (SSL) techniques, graph-based approaches have received superior attention due to their uniqueness of structure, universality of applications, and scalability to large scale data. In fact, many data sets, like social networks, can be represented by graphs. Undirected graphs also make it easier to formulate the learning problem into a convex optimization problem, which can be solved with existing techniques. Most importantly, the expressive power of graph structure under the manifold assumption in semi-supervised learning contributes to the success of graph-based semi-supervised methods. Specifically, the similarity graph construction implies that vertices connected by an edge associated with a large weight tend to have the same label, which coincides to the manifold assumption of SSL.

Overall, due to the integration of several aforementioned advantageous techniques, the proposed method demonstrates its benefits most prominently in the case of a small amount of labeled samples; this scenario is common since labeled data is scarce for many applications. This common case, however, is a challenge for many machine learning methods since they often require a large amount of labeled samples to learn an accurate model, especially if unlabeled data is not used during training. Among graph-based semi-supervised algorithms, our method has the additional benefit of integrating persistent spectral graph theory, which has been shown to improve accuracy of predictions in ablation studies of Table 4.

The contributions of the paper are summarized as follows:

- We present a new algorithm, called persistent Laplacian-enhanced graph MBO (PL-MBO), for data classification for low label rates or cases of low amounts of labeled data.
- The proposed algorithm uses a family of persistent Laplacian matrices to obtain topological features of data sets that persist across multiple scales, thus giving the algorithm valuable information.

- The proposed method requires a reduced amount of labeled data for accurate classification compared to many other machine learning methods. In fact, it works well even with very low amounts of labeled data, which is important due to the scarcity of labeled data.
- The proposed algorithm is very efficient.
- The proposed method can be adapted for both small and large data sets, as outlined in Sect. 3.2, and works well for both types of data.

The remainder of the paper is organized as follows: in Sect. 2, we present background information on related work, the graph framework, and persistent Laplacians. In Sect. 3, we present the graph MBO technique and derive our proposed method. The results of the experiments on benchmark data sets and the discussion of the results are presented in Sect. 4. Section 5 provides concluding remarks.

2 Background

2.1 Related work

Here, we review recent graph-based methods and persistent Laplacian-related algorithms, with a focus on semi-supervised techniques. In particular, graph-based methods usually utilize either the transductive or the inductive setting. The goal of the transductive setting is to predict the class of the set of unlabeled elements, while the goal of the inductive setting is to learn a function that can classify any element.

Label propagation is one of the earliest and yet popular methods for the label inference task. Some of the earliest yet popular methods are the Gaussian random fields method (Zhu and Ghahramani, 2002), the local and global consistency method (Zhou et al., 2003), special label propagation (Nie et al., 2010) and the linear neighborhood propagation method (Wang and Zhang, 2006). Some general label propagation regularization-based methods include directed regularization (Zhou et al., 2005), manifold regularization (Belkin et al., 2006; Xu et al., 2010), anchor graph regularization (Liu et al., 2010), label propagation algorithm via deformed graph Laplacians (Gong et al., 2015), Tikhonov regularization (Belkin et al., 2004b), and interpolated regularization (Belkin et al., 2004a).

Some recently developed label propagation methods involve adaptations of the original Merriman–Bence–Osher (MBO) scheme (Merriman et al., 1994), which is an efficient numerical algorithm for approximating motion by mean curvature, to different tasks. In particular, Merkurjev et al. (2013) introduces a graphical MBO-scheme based algorithm for segmentation and image processing, while Garcia-Cardona et al. (2014), Merkurjev et al. (2014a) present a fast multiclass segmentation technique using diffuse interface methods on graphs. Moreover, Meng et al. (2017), Merkurjev et al. (2014b) develop algorithms for hyperspectral imagery, while (Merkurjev, 2020) presents a new graph-based method for the unsupervised classification of 3D point clouds, and Merkurjev et al. (2018) incorporates heat kernel pagerank and variations of the MBO scheme. Additionally, Jacobs et al. (2018) develops a new auction dynamics framework for data classification, which is able to integrate class size information and volume constraints. Furthermore, Merkurjev et al. (2022) introduces a multiscale graph-based MBO scheme that incorporates multiscale graph Laplacians and adaptations of the classical MBO scheme (Merriman et al., 1994). Lastly, Hayes et al. (2023) develops three graph-based methods for the prediction of scarcely labeled molecular data by integrating transformer and autoencoder techniques with the classical MBO procedure.

Other popular graph-based semi-supervised methods include shallow graph embedding algorithms. In particular, shallow graph embedding techniques include factorization-based algorithms such as locally linear embedding (Roweis and Saul, 2000), Laplacian eigenmaps (Belkin and Niyogi, 2001), the graph factorization algorithm (Ahmed et al., 2013), GraRep (Cao et al., 2015), and HOPE (Ou et al., 2016). Moreover, a subset of graph embedding techniques includes those which incorporate random walks; some random walk-based algorithms include: DeepWalk (Perozzi et al., 2014), Planetoid (Yang et al., 2016), Node2Vec (Grover and Leskovec, 2016), LINE (Tang et al., 2015), and HARP (Chen et al., 2018). In addition, recently many deep embedding approaches have been proposed. A few of autoencoder-based methods include deep neural networks for learning graph representation (DNGR), Cao et al. (2016), deep recursive network embedding (DRNE) (Tu et al., 2018), and structural deep network integration (SDNE) (Wang et al., 2016).

In addition, the success of convolutional neural networks (CNNs) has led to many adaptations of CNNs for graph-based and semi-supervised frameworks. In particular, Kipf and Welling's seminal work (Kipf and Welling, 2017) proposes a semi-supervised graph convolutional network (GCN), where the convolutional architecture is developed via a localized first-order approximation of spectral graph convolutions. In addition, Li et al. (2018) proposes an adaptive graph convolutional network by constructing a residual graph using a learnable distance function with two-node features as input. In recent years, variants of graph neural networks (GNNs), such as the graph attention network described in Velickovic et al. (2018); Zhang et al. (2018); Wang et al. (2019), have been developed and shown great success in deep learning tasks and problems. For more information on graph-based methods, one can refer to the review papers (Song et al., 2022; Van Engelen and Hoos, 2020).

Other recent semi-supervised graph based methods include the Centered Kernel method (Mai and Couillet, 2018), a p-Laplace learning technique (Flores et al., 2022), Poisson learning, Calder et al. (2020) Poisson MBO method, Calder et al. (2020) Dynamic Label Propagation method, Wang et al. (2013), Sparse label Propagation method (Jung et al., 2016), Semi-supervised learning by the Absolutely Minimal Lipschitz Extension (AMLE) (Bungert et al., 2023), Semi-supervised learning by via the solution of the graph 'graph.peikonal' equation (Calder and Ettehad, 2022). Our paper also presents a graph-based semi-supervised learning framework for cases of low amounts of labeled data and uses Laplacian-based techniques as well, except we integrate spectral graph theory to construct a family of persistent Laplacians; we then incorporate threshold dynamics techniques into our proposed procedure.

Moreover, other recent graph-based methods include graph inference learning (GIL) (Xu et al., 2020), where the authors define a structural relation that combines node attributes, paths between nodes, and local topological structures to establish a connection between two nodes, Wang et al. (2021), where the authors propose label propagation with structured graph learning (LPSGl) methods for semi-supervised learning, and Cai et al. (2021), where the authors propose fully linear graph convolutional networks (FLGC) for semi-supervised learning and clustering, where they train FLGC based on computing a globally optimal closed-form solution with a decoupled procedure, resulting in a generalized linear framework. Other recent methods that integrate graph convolutional networks include semi-supervised classification by graph p-Laplacian convolutional networks (GpLCN) (Fu et al., 2021b), graph Laplacian regularized graph convolutional networks (Jiang and Lin, 2018), dynamic graph learning convolutional networks (DGLCN) (Fu et al., 2021a), graph convolutional networks using heat kernel (GraphHeat) (Xu et al., 2020), as well as variance-enlarged graph Poisson networks (Zhou et al., 2023).

In addition to the above, it is important to note that the proposed method utilizes topological tools from persistent spectral graph theory. In particular, while persistent homology

is a powerful tool in topological data analysis (TDA) for investigating the structure of data (Edelsbrunner and Harer, 2008; Zomorodian and Carlsson, 2004), it is incapable of describing the homotopic shape evolution of data during filtration. Therefore, in Wang et al. (2020), the authors introduce persistent spectral theory to extract rich topological and spectral information of data through a filtration process, while the theoretical properties of persistent Laplacians are presented in Mémoli et al. (2022). Overall, persistent Laplacians have had tremendous success in computational biology and biophysics, such as protein-ligand binding (Meng and Xia, 2021), protein-protein binding problems (Chen et al., 2022), and protein thermal stability (Wang et al., 2020). Motivated by the success of topological persistence, we integrate persistent spectral tools to develop our proposed method. Specifically, in this paper, we integrate adaptations of the MBO scheme with persistent Laplacian techniques to develop a semi-supervised graph-based method for data classification that can perform well in cases of data with few labeled elements.

2.2 Graph-based framework

In this section, we review the graph-based framework used in this paper. Specifically, let $G = (V, E)$ be an undirected graph, where V and E are the sets of vertices and edges, respectively. The vertex set $V = \{x_1, \ldots, x_N\}$ is associated with the elements of the data and E is the set of edges connecting pairs of vertices. The similarity between vertices x_i and x_j is measured by a weight function $w : V \times V \to \mathbb{R}$. The weight function values are usually in the interval $[0, 1]$ and are equipped with the following property: a large value of $w(x_i, x_j)$ indicates that vertices x_i and x_j are similar to each other, whereas a small value indicates they are dissimilar; thus, the graph-based framework is able to provide crucial information about the data. The weight function is also symmetric. Some popular weight functions include:

- the Gaussian weight function

$$w(x_i, x_j) = \exp\left(-\frac{d(x_i, x_j)^2}{\sigma^2}\right), \tag{1}$$

  where $d(x_i, x_j)$ represents a distance (computed using a measure) between vertices x_i and x_j, associated with the ith and jth data elements, and $\sigma > 0$ is a parameter which controls scaling in the weight function.
- Zelnik-Manor and Perona (ZMP) weight function (Zelnik-Manor and Perona, 2004)

$$w(x_i, x_j) = \exp\left(-\frac{d(x_i, x_j)^2}{\sigma(x)\sigma(y)}\right), \tag{2}$$

  where $d(x_i, x_j)$ represents a distance metric and $\sqrt{\sigma(x_i)} = d(x_i, x_M)$ is a local parameter for each x_i, where x_M is the Mth closest vector to x_i.
- Cosine similarity weight function (Singhal, 2001)

$$w(x_i, x_j) = \cos(x_i, x_j) = \frac{\langle x_i, x_j \rangle}{||x_i||||x_j||} \tag{3}$$

Overall, there are a few important terms to define in a graph-based framework. For example, the degree of vertex $x_i \in V$ is defined as

$$d(x_i) = \sum_j w(x_i, x_j).$$

Moreover, denote $\mathbf{W}$ as the weight matrix $\mathbf{W}_{i,j} = w(x_i, x_j)$. If $\mathbf{D}$ is the diagonal matrix with the degrees of the vertices as elements, then we can define the graph Laplacian as $\mathbf{L} = \mathbf{D} - \mathbf{W}$. In some cases, the graph Laplacian is normalized to account for the behavior that arises when the sample size is large. One example of a normalized graph Laplacian is the symmetric graph Laplacian defined as:

$$\mathbf{L_s} = \mathbf{I} - \mathbf{D^{-1/2}WD^{-1/2}}. \tag{4}$$

In this work, to derive our proposed method, we use a filtration procedure to generate a sequence of chain complexes and associated families of simplicial complexes and chain complexes, from which we construct a family of persistent Laplacians, which allows us to capture important information from the data. In the next section, we review some background on persistent Laplacians.

2.3 Persistent Laplacians

In this section, we review some basic notions to formulate the persistent Laplacian matrix on the simplicial complex. The details can be found in Wang et al. (2020).

2.3.1 Simplicial complex

For $q \geq 0$, a $q-$simplex σ_q in an Euclidean space $\mathbb{R}^n$ is the convex hall of a set P of $q+1$ affinely independent points in $\mathbb{R}^n$. In particular, a 0-simplex is a vertex, a 1-simplex is an edge, a 2-simplex is a triangle, and a 3-simplex is a tetrahedron. A q-simplex is said to have dimension q.

Moreover, a simplicial complex K is a (finite) collection of simplices in $\mathbb{R}^n$ such that

1. Every face (σ_p) of a simplex of K is in K.
2. The non-empty intersection of any two simplices of K is a face of each.

2.3.2 Chain complex

Let K be a simplicial complex of dimension q. A q-chain is a formal sum of q-simplices added with some coefficients. Under the addition operation of $\mathbf{Z}_2$, a set of all q-chains forms a chain group $C_q(K)$. One can also relate chain groups at different dimensions by a boundary operator: given a q-simplex $\sigma_q = \{v_0, \ldots, v_q\}$, we define the boundary operator $\partial_q^K : C_q(K) \to C_{q-1}(K)$ by

$$\partial_q \sigma_q = \sum_{i=0}^{q} \{v_0, \ldots, \hat{v_i}, \ldots, v_q\}, \tag{5}$$

where $\hat{v_i}$ indicates the vertex v_i is omitted. In general, the boundary operator changes a q-simplex to a $(q-1)$-simplex.

A chain complex is a sequence of chain groups connected by boundary operators. Similar to boundaries of chains, we have the notion of coboundaries of cochains defined as

$$\partial_q^* : C_{q-1}(K) \to C_q(K). \tag{6}$$

Moreover, the qth homology group of K is $H_q(K) = Z_q(K)/B_q(K)$, where $Z_q(K)$ is $ker(\partial_q^K)$ and $B_q(K)$ is $im(\partial_{q+1}^K)$. In addition, the qth Betti number is the rank of the

q-dimensional homology: $\beta_q^K = rank(H_q(K))$; the Betti number reveals the intrinsic topological information of a geometry or data. Specifically, β_0^t provides the number of connected components in K_t, β_1^t provides the number of one-dimensional circles and β_2^t gives the number of two-dimensional voids in K_t. For K oriented simplicial complex, for $q \geq 0$, the q-combinatorial Laplacian is a linear operator that maps $C_q(K)$ to $C_q(K)$:

$$\Delta_q := \partial_{q+1}\partial_{q+1}^* + \partial_q^*\partial_q. \tag{7}$$

Similarly, we can represent the q-combinatorial Laplacian matrix as

$$\mathcal{L}_q = \mathcal{B}_{q+1}\mathcal{B}_{q+1}^T + \mathcal{B}_q^T\mathcal{B}_q, \tag{8}$$

where $\mathcal{B}_q$ and $\mathcal{B}_q^T$ is the matrix representation of the q-boundary operator and the q-coboundary operator $\partial_q^* : C_{q-1}(K) \to C_q K$ defined in Hatcher (2005), respectively. Note that when K is a graph, we have $\mathcal{L}_0(K) = \mathcal{B}_1\mathcal{B}_1^T + \mathcal{B}_0^T\mathcal{B}_0$. This means that the 0-combinatorial Laplacian matrix is similar to a graph Laplacian matrix defined in the Sect. 2.2 except that their matrix values are very different. Note that the use of simplicial complexes allows high-dimensional modeling in combinatorial Laplacians, whereas graph Laplacians can only support pairwise interactions.

2.3.3 Filtration

The notion of filtration is at the core of topological persistence. In particular, a filtration can be defined in the context of topological spaces or simplicial complexes. Specifically, a filtration of $\mathcal{F} = \mathcal{F}(K)$ of a oriented simplicial complex K is a nested sequence of its subcomplexes

$$\mathcal{F} : \phi = K_0 \subseteq K_1 \subseteq \cdots \subseteq K_n = K. \tag{9}$$

Overall, it induces a sequence of chain complexes:

$$\begin{array}{ccccccccccccc}
\cdots & C_{q+1}^1 & \underset{\partial_{q+1}^{1*}}{\overset{\partial_{q+1}^1}{\rightleftarrows}} & C_q^1 & \underset{\partial_q^{1*}}{\overset{\partial_q^1}{\rightleftarrows}} & \cdots & \underset{\partial_3^{1*}}{\overset{\partial_3^1}{\rightleftarrows}} & C_2^1 & \underset{\partial_2^{1*}}{\overset{\partial_2^1}{\rightleftarrows}} & C_1^1 & \underset{\partial_1^{1*}}{\overset{\partial_1^1}{\rightleftarrows}} & C_0^1 & \underset{\partial_0^{1*}}{\overset{\partial_0^1}{\rightleftarrows}} & C_{-1}^1 \\
 & \cap & & \cap & & & & \cap & & \cap & & \cap & & \\
\cdots & C_{q+1}^2 & \underset{\partial_{q+1}^{2*}}{\overset{\partial_{q+1}^2}{\rightleftarrows}} & C_q^2 & \underset{\partial_q^{2*}}{\overset{\partial_q^2}{\rightleftarrows}} & \cdots & \underset{\partial_3^{2*}}{\overset{\partial_3^2}{\rightleftarrows}} & C_2^2 & \underset{\partial_2^{2*}}{\overset{\partial_2^2}{\rightleftarrows}} & C_1^2 & \underset{\partial_1^{2*}}{\overset{\partial_1^2}{\rightleftarrows}} & C_0^2 & \underset{\partial_0^{2*}}{\overset{\partial_0^2}{\rightleftarrows}} & C_{-1}^1 \\
 & \cap & & \cap & & & & \cap & & \cap & & \cap & & \\
 & \vdots & & \vdots & & & & \vdots & & \vdots & & \vdots & & \\
 & \cap & & \cap & & & & \cap & & \cap & & \cap & & \\
\cdots & C_{q+1}^m & \underset{\partial_{q+1}^{m*}}{\overset{\partial_{q+1}^m}{\rightleftarrows}} & C_q^m & \underset{\partial_q^{m*}}{\overset{\partial_q^m}{\rightleftarrows}} & \cdots & \underset{\partial_3^{m*}}{\overset{\partial_3^m}{\rightleftarrows}} & C_2^m & \underset{\partial_2^{m*}}{\overset{\partial_2^m}{\rightleftarrows}} & C_1^m & \underset{\partial_1^{m*}}{\overset{\partial_1^m}{\rightleftarrows}} & C_0^m & \underset{\partial_0^{m*}}{\overset{\partial_0^m}{\rightleftarrows}} & C_{-1}^1,
\end{array} \tag{10}$$

where $C_q^t := C_q(K_t)$ and $\partial_q^t : C_q(K_t) \to C_{q-1}(K_t)$.

2.3.4 Persistent Laplacians

At the core of our proposed method is the persistent Laplacian matrix. In this section, we discuss the concept in more detail. Consider $\mathbb{C}_q^{t+p}$, the subset of C_q^{t+p} whose boundary is in

C^t_{q-1}, defined by:

$$\mathbb{C}^{t+p}_q = \{e \in C^{t+p}_q | \partial^{t+p}_q(e) \in C^t_{q-1}\} \subseteq C^{t+p}_q. \tag{11}$$

We define the p-persistent q-boundary operator as $\eth^{t+p}_q : \mathbb{C}^{t+p}_q \to C^t_{q-1}$ and the adjoint boundary operator as $(\eth^{t+p}_q)^* : C^t_{q-1} \to \mathbb{C}^{t+p}_q$; both operators are well-defined. The p-persistent q-combinatorial Laplacian operator is defined as:

$$\Delta^{t+p}_q = \eth^{t+p}_{q+1}(\eth^{t+p}_{q+1})^* + (\partial^t_q)^* \partial^t_q.$$

Now, we denote the matrix representation of $\eth^{t+p}_{q+1}$ and $\eth^t_q$ by $\mathcal{B}^{t+p}_{q+1}$ and $\mathcal{B}^t_q$, respectively. Similarly, we can represent $(\eth^{t+p}_{q+1})^*$ and $(\eth^t_q)^*$ by matrices $(\mathcal{B}^{t+p}_{q+1})^T$ and $(\mathcal{B}^t_q)^T$, respectively. Therefore, the p-persistent q-combinatorial Laplacian matrix is defined as:

$$\mathcal{L}^{t+p}_q = \mathcal{B}^{t+p}_{q+1}(\mathcal{B}^{t+p}_{q+1})^T + (\mathcal{B}^t_q)^T \mathcal{B}^t_q. \tag{12}$$

We also denote the set of spectral of $\mathcal{L}^{t+p}_q$ by

$$\text{Spectra}(\mathcal{L}^{t+p}_q) = \{(\lambda_1)^{t+p}_q, (\lambda_2)^{t+p}_q, \cdots, (\lambda_N)^{t+p}_q\},$$

where the spectra are arranged in ascending order. Moreover, the p-persistent qth Betti numbers can be defined as the number of zero eigenvalues of the p-persistent q-combinatorial Laplacian matrix $\mathcal{L}^{t+p}_q$. Thus,

$$\beta^{t+p}_q = \dim(\mathcal{L}^{t+p}_q) - \text{rank}(\mathcal{L}^{t+p}_q) = \text{nullity}(\mathcal{L}^{t+p}_q) = \#\text{of zero eigenvalues of} \mathcal{L}^{t+p}_q.$$

In the above definition, β^{t+p}_q counts the number of q-cycles in K_t that are still alive in K_{t+p}. This topological information is exactly what can be obtained using persistent homology. However, persistent spectral theory provides additional geometric information using the spectra of persistent combinatorial Laplacians. More specifically, the non-harmonic spectra can capture both the topological changes and the homotopic shape evaluation of the data; see Fig. 1 for an illustration. More detailed descriptions of persistent spectral graphs can be found in Wang et al. (2020).

3 Methods

This section provides the graph MBO technique and its LP generalization.

3.1 The graph MBO technique

The proposed data classification method derived in this paper is based on the techniques outlined in the literature, such as Garcia-Cardona et al. (2014), Merkurjev et al. (2014a, 2013), Hayes et al. (2023), Merkurjev et al. (2022), which generalize the original MBO algorithm (Merriman et al., 1994) into a graphical setting.

For the derivation of our method, consider a matrix $\mathbf{U} = (u_1, \ldots, u_N)^T \in \mathbb{R}^{N,K}$, where K is the number of classes, N is the number of data elements and $u_i \in \mathbb{R}^K$ indicates the probability distribution over the different classes for the data element x_i; thus, the jth element of u_i is the probability of data element x_i of belonging to class j. In particular, the vector u_i

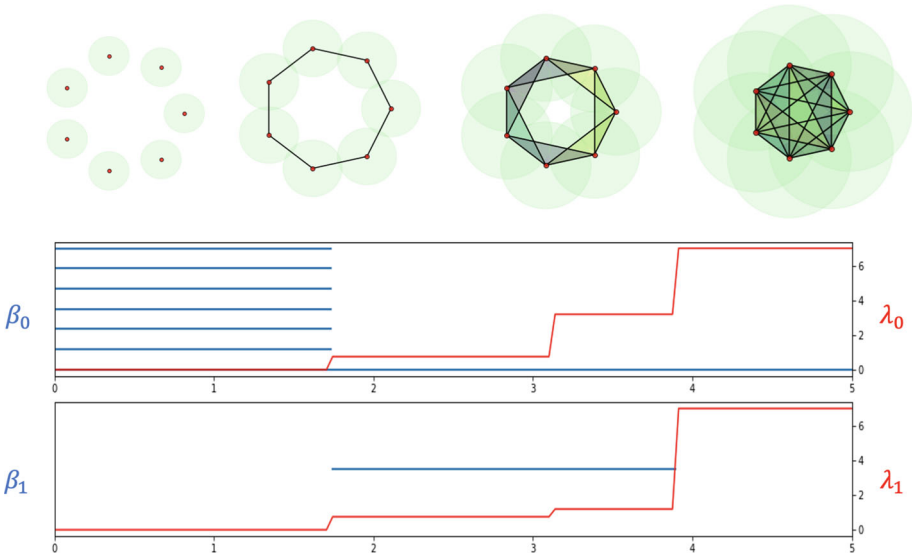

Fig. 1 Comparison of persistent homology and persistent Laplacians. The top panel shows the same data (seven points) at four different stages of filtration characterized by a radius r. The second and third panels represent the corresponding topological (i.e., harmonic) and non-harmonic features of dimension 0 and dimension 1, respectively. The blue line represents the persistent homology (PH) barcodes of dimension 0 ($\beta_0(r)$) and dimension 1 ($\beta_1(r)$), while the red line represents the first non-zero eigenvalues of dimension 0 ($\lambda_0(r)$) and dimension 1 ($\lambda_1(r)$) of the persistent Laplacians (PLs). It is shown that the harmonic spectra of PLs return all topological invariants of PH and the non-harmonic spectra of PLs reveal the additional homotopic shape evolution of PLs during the filtration (i.e., the second jumps in the red curves correspond to the increase in connectivity, the third geometric shape, but there was no topological change). We note that persistent homology fails to capture the homotopic shape evolution during filtration. The details on persistent spectral graphs can be found in the work (Wang et al., 2020) (color figure online)

is an element of the Gibbs simplex Σ^K:

$$\Sigma^K := \{(x_1, \ldots, x_K) \in [0,1]^K \mid \sum_{k=1}^{K} x_k = 1\}. \tag{13}$$

The goal of the proposed technique will be to compute the optimal matrix $\mathbf{U}$.

Regarding the data classification problem of machine learning, many data classification algorithms can be regarded as optimization techniques where a specific objective function is optimized. In particular, one of the popular optimization techniques for data classification involves minimizing the following general objective function:

$$E(\mathbf{U}) = R(\mathbf{U}) + F(\mathbf{U}), \tag{14}$$

where $\mathbf{U}$ is a classification variable with each row representing a probability distribution of a particular data element over the classes, $R(\mathbf{U})$ is a regularization term and $F(\mathbf{U})$ is a fidelity term containing information about labeled data. The goal is to minimize $E(\mathbf{U})$ and thus to obtain the optimal $\mathbf{U}$.

Regarding the regularization term $R(\mathbf{U})$, it may be useful to examine a certain functional called the Ginburg-Landau (GL) functional, described in more detail in Merkurjev et al.

(2013). The classical GL functional takes the following form:

$$\mathrm{GL}(u) = \frac{\epsilon}{2}\int |\nabla u|^2 dx + \frac{1}{\epsilon}\int W(u)dx, \tag{15}$$

where u is a scalar field representing the state of the phases in the system, $W(u)$ is a double-well potential, ϵ is a positive constant and ∇ denotes the spatial gradient operator. In Bertozzi and Flenner (2012), Garcia-Cardona et al. (2014), the authors modify the original Ginzburg-Landau functional (15) into a graph-based functional by replacing the first term with the graph Dirichlet energy to obtain a graph-based regularization term. In addition, to incorporate multiple classes, they modify the double-well potential into a multi-class setting; one can also add an $\mathbf{L}^2$ penalty term to incorporate the labels of the labeled elements. For more details about this graph-based representation of the Ginzburg-Landau functional, one can refer to Garcia-Cardona et al. (2014), Merkurjev et al. (2014a, 2013).

Inspired by the above, the optimization problem we consider in this work consists of minimizing the following graph-based Ginzburg-Landau energy:

$$E(\mathbf{U}) = \frac{\epsilon}{2}\langle \mathbf{U}, \mathbf{L_{norm}}\mathbf{U}\rangle + \frac{1}{2\epsilon}\sum_{i\in V}(\prod_{k=1}^{K}\frac{1}{4}||u_i - e_k||_{L_1}^2) + \sum_{i\in V}\frac{\mu_i}{2}||u_i - \hat{u}||^2, \tag{16}$$

where $\langle \mathbf{U}, \mathbf{L_{norm}}\mathbf{U}\rangle$ = trace $(\mathbf{U}^T\mathbf{L_{norm}}\mathbf{U})$, $\mathbf{L_{norm}}$ is any normalized graph Laplacian such as the symmetric graph Laplacian, K is number of classes, and u_i is the ith row of $\mathbf{U}$. In addition, $\hat{u_i}$ is a vector indicating the prior class knowledge of x_i, e_k is an indicator vector of size K with a one in kth component and zero elsewhere, and μ_i takes some positive value if x_i is labeled data element and 0 otherwise.

To minimize the graph-based multiclass energy functional in (16), the authors of Garcia-Cardona et al. (2014) developed a convex splitting scheme. Similarly, the authors of Merkurjev et al. (2013) derived a modified MBO scheme to minimize (16). More specifically, the authors drew upon a technique that can be used to minimize the classical non-graphical GL functional (15), which can be optimized in the $\mathbf{L}_2$ sense using gradient descent, resulting in an Allen–Cahn equation. If a time-splitting scheme is then applied, one obtains a procedure where one alternates between propagation using the heat equation with a forcing term and thresholding, which is similar to the steps of the original MBO scheme (Merriman et al., 1994), an efficient numerical technique for computing an approximation of mean curvature flow. This can be extended to a graphical and multiclass setting by using a graph Laplacian and projecting to the closest vertex in the Gibbs simplex (13). For a detailed explanation, one can refer to the work (Merkurjev et al., 2013).

Overall, the goal of this paper is to integrate similar techniques with persistent spectral graphs and a filtration procedure, from which we construct a family of persistent Laplacians. Incorporating persistent Laplacians into our proposed procedure will allow us to obtain crucial information about the data, such as all topological invariants and the homotopic shape evolution of the data during the filtration.

3.2 PL-MBO algorithm

In this section, we will derive our proposed semi-supervised method, PL-MBO, for data classification, which is especially useful for cases with low label rates. To derive the proposed method, the PL-MBO algorithm, we will consider minimizing a variant of (16), where instead of choosing the graph Laplacian to be the normalized graph Laplacian, we will choose it from a family of persistent Laplacians.

Consider a simple graph; we can then generate persistent Laplacians from a weighted graph Laplacian by using a threshold in the matrix computation. In particular, we define the weighted Laplacian as $\mathbf{L_{norm}} = (L_{ij})$, where $L_{ii} = -\sum_{j=1}^{N} L_{ij}$ and $L_{ij} \leq 0$ for all i and j, and N is the number of data elements in the data set.

For $i \neq j$, let $L_{\max} = \max_{ij} L_{ij}$, $L_{\min} = \min_{ij} L_{ij}$, and $d = L_{\max} - L_{\min}$.

Let L_n be an integer greater than 1. We can then define the kth persistent Laplacian, $\mathbf{L}^k_{\text{persistent}}$, for $k = 1, 2, \ldots, L_n$, as $(\mathbf{L}^k_{\text{persistent}})_{ij} = L^k_{ij}$, where, for $i \neq j$:

$$L^k_{ij} = \begin{cases} 0 & \text{if } L_{ij} \leq \frac{k}{L_n} d + L_{\min}, \\ -1 & \text{otherwise.} \end{cases} \tag{17}$$

The diagonal entries of the persistent Laplacian, $\mathbf{L}^k_{\text{persistent}}$, are computed as

$$L^k_{ii} = -\sum_{j=1}^{N} L^k_{ij}. \tag{18}$$

In our proposed method, $\{\mathbf{L}^k_{\text{persistent}}\}$ (for $k = 1, 2, \ldots, L_n$) are used in place of $\mathbf{L_{norm}}$ in (16). Specifically, a variant of (16) with each derived persistent Laplacian from the family is minimized using similar techniques to those described at the end of Sect. 3.1. Finally, the results from each persistent Laplacian assisted MBO technique are concatenated and fed into a classifier, such as a gradient-boosting decision tree, support vector machine, a random forest, or another classifier, which predicts the final class of each data element.

In particular, let $\mathbf{U}$ represent a matrix where each row u_i contains the probability distribution of each data element over the classes. Also, let $dt > 0$ be the step size, N be the number of data elements, and K be the number of classes. Moreover, let the vector $\boldsymbol{\mu}$ represent a vector that takes the value μ at labeled data elements and 0 at unlabeled data elements. In addition, we define the $\mathbf{U_{labeled}}$ matrix as follows: for labeled elements, each row is an indicator vector with a 1 in the entry corresponding to the class of the labeled element. All other entries are set to 0.

We can summarize our proposed method as follows:

- Using the input data, construct a similarity graph using a chosen similarity function such as (2), and then compute the symmetric graph Laplacian (4).
- Using the symmetric graph Laplacian, construct a family of L_n persistent Laplacians, i.e. $\{\mathbf{L}^k_{\text{persistent}}\}$, for $k = 1, 2, \ldots, L_n$, where L_n is an integer greater than one, as derived in Sect. 2.3.4.
- Use a spectral technique: in particular, for each persistent Laplacian in the family, i.e. $\mathbf{L}^k_{\text{persistent}}$, for $k = 1, 2, \ldots, L_n$, compute the smallest N_e eigenvalues and associated eigenvectors. It is important to note that usually only a small portion of the eigenvalues and associated eigenvectors are needed to be computed for accurate predictions. Please see next page for more detail.
- Initialize $\mathbf{U}$ using techniques such as random initialization or Voronoi initialization. In particular, in Voronoi initialization, the labels of the unlabeled points are initialized by creating a Voronoi diagram with the labels of the labeled points as the seed points; every point is assigned the label of the labeled point in its Voronoi cell. We note that, in any initialization, the rows of the initial $\mathbf{U}$ corresponding to labeled points should consist of indicator vectors with a 1 at the place corresponding to the true class of the data element.

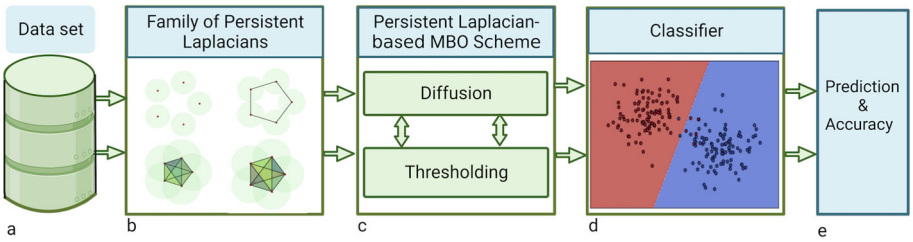

Fig. 2 Visual description of PL-MBO. **a** From a given data set, a similarity graph is constructed using a chosen similarity function. **b** A family of persistent Laplacians is formed as derived in Sect. 2.3. **c** The graph-based Ginzburg-landau energy (16) is minimized using a modified persistent Laplacian-based MBO scheme incorporating the family of persistent Laplacians; a new test set is formed using the output. **d** A machine learning algorithm is used to classify the new test data. **e** The accuracy of the proposed method is computed on the test data set

- For each persistent Laplacian in the family, i.e. $\mathbf{L}^k_{\text{persistent}}$, for $k = 1, 2, \ldots, L_n$, perform the following MBO-like steps for N_t iterations, to obtain the next iterate of $\mathbf{U}$; if there are L_n persistent Laplacians, there will be L_n output matrices $\mathbf{U}$:
  1. Heat equation with a forcing term:
  $$\mathbf{U}^{n+\frac{1}{2}} = \mathbf{U}^n - dt\{\mathbf{L}^k_{\text{persistent}}\mathbf{U}^{n+\frac{1}{2}} + \boldsymbol{\mu} \cdot (\mathbf{U}^n - \mathbf{U}_{\mathbf{labeled}})\},$$
  where $\boldsymbol{\mu}$ is a vector which takes a value μ in the ith place if $\mathbf{x}_i$ is a labeled element and 0 otherwise, and the term $\boldsymbol{\mu} \cdot (\mathbf{U}^n - \mathbf{U}_{\mathbf{labeled}})$ indicates row-wise multiplication by a scalar. Later, we describe the spectral techniques used to make the first step efficient even for larger data sets.
  2. Projection to simplex: Each row of $\mathbf{U}^{n+\frac{1}{2}}$ is projected onto the simplex using Chen and Ye (2011).
  3. Displacement: $u_i^{n+1} = \boldsymbol{e}_k$, where u_i^{n+1} is the ith row of $\mathbf{U}^{n+1}$, and $\boldsymbol{e}_k$ is the indicator vector.
- Concatenate the results of each output matrix (from each persistent Laplacian) to form a new matrix. For the binary case, one only needs to concatenate the first column of each output matrix to form the new matrix.
- Use a classifier, such as a gradient-boosting decision tree, a support vector machine or a random forest, to predict the final class of the data elements. The rows corresponding to $\boldsymbol{\mu_i} = \mu$ are used for training of the classifier.

The proposed PL-MBO procedure is detailed as Algorithm 1. For an illustration, an intuitive interpretation of the proposed method is shown in Fig. 2.

To make the scheme even more efficient, we use a spectral technique and utilize a low-dimensional subspace spanned by a small number of eigenfunctions, similar to the procedures outlined in Merkurjev et al. (2014a, 2013), Garcia-Cardona et al. (2014). The idea is to first rewrite Step 1 (Heat equation with a forcing term) as

$$\mathbf{U}^{n+\frac{1}{2}} = \mathbf{C}^{-1}\mathbf{U}_{\mathbf{update}}, \tag{19}$$

where $\mathbf{C} = \mathbf{I} + dt\mathbf{L}^k_{\text{persistent}}$ and $\mathbf{U}_{\mathbf{update}} = \mathbf{U}^{\mathbf{n}} - dt\boldsymbol{\mu} \cdot (\mathbf{U}^n - \mathbf{U}_{\mathbf{labeled}})$.

Now, let the eigendecomposion of $\mathbf{L}^k_{\text{persistent}}$ be denoted as $\mathbf{L}^k_{\text{persistent}} = \mathbf{X}\Lambda\mathbf{X}^{\mathbf{T}}$, and let $\mathbf{X}_{\mathbf{truncated}}$ and $\Lambda_{\mathbf{truncated}}$ be truncated matrices of $\mathbf{X}$ and Λ, respectively, containing only N_e smallest eigenvectors and eigenvalues of the persistent Laplacian, respectively, where

Algorithm 1: PL-MBO Algorithm

Require: N (# of data set elements), m (# of labeled data elements), labeled data $\mathcal{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{m}$, where y_i is the label of $\mathbf{x}_i$, unlabeled data $\mathcal{U} = \{\mathbf{x}_j\}_{j=m+1}^{N}$, N_n (# of nearest neighbors), $\sigma > 0$, $dt > 0$, $N_e << N$ (# of eigenvectors to be computed), N_t (maximum # of iterations), $\boldsymbol{\mu}$ (an $N \times 1$ vector which takes a value μ in the i^{th} place if $\mathbf{x}_i$ is a labeled element and 0 otherwise), L_n (# of persistent Laplacians).

Ensure: Prediction of the final class for each data element.

1: Construct a N_n-nearest neighbors graph from the data.

2: Compute the symmetric graph Laplacian (4).

3: Construct a family of L_n persistent Laplacians from the graph Laplacian using (17) and (18).

4: Compute the matrices $\mathbf{U_{labeled}}$, $\mathbf{\Lambda_{truncated}}$ and $\mathbf{X_{truncated}}$ for each persistent Laplacian as described in Section 3.2. Let $\mathbf{X_p} = \mathbf{X_{truncated}}$ for $\mathbf{L^p_{persistent}}$ and $\mathbf{\Lambda_p} = \mathbf{\Lambda_{truncated}}$ for $\mathbf{L^p_{persistent}}$.

5: Complete the following steps starting with $n = 1$:

for $i = 1 \rightarrow N$ and all k **do**
  $\mathbf{U}^0_{ik} \leftarrow rand((0, 1))$
  $\mathbf{u}^0_i \leftarrow projectToSimplex(\mathbf{u}^0_i)$ using (Chen and Ye, 2011), where $\mathbf{u}^0_i$ is i^{th} row of $\mathbf{U}^0$.
  If $\boldsymbol{\mu}_i > 0$, $\mathbf{U}^0_{ik} \leftarrow \mathbf{U}_{\text{labeled}\,ik}$
end for
for $p = 1 \rightarrow L_n$ **do**
  $\mathbf{B_p} \leftarrow (\mathbf{I} + \mathrm{dt}\boldsymbol{\Lambda_p})^{-1}\,\mathbf{X_p}^T$
  while $n < N_t$ **do**
    $\mathbf{F} \leftarrow \mathbf{U}^n - \mathrm{dt}\,\boldsymbol{\mu} \cdot (\mathbf{U}^n - \mathbf{U}_{\text{labeled}})$
    $\mathbf{A_p} \leftarrow \mathbf{B_p F}$
    $\mathbf{U_p}^{n+1} \leftarrow \mathbf{X_p A_p}$
    for $i = 1 \rightarrow N$ **do**
      $\mathbf{v}_i^{n+1} \leftarrow projectToSimplex(\mathbf{u}_i^{n+1})$ using (Chen and Ye, 2011)
      $\mathbf{u}_i^{n+1} \leftarrow \boldsymbol{e}_k$, where k is closest simplex vertex to $\mathbf{v}_i^{n+1}$
    end for
    The matrix $\mathbf{U_p}^{n+1}$ is such that its i^{th} row is $\mathbf{v}_i^{n+1}$.
    $n \leftarrow n + 1$
  end while
  $\mathbf{X} = Concatenate\{\mathbf{U_p}_{p=1}^{L_n}\}$
end for
$j, k = 1$
for $i = 1 \rightarrow N$ **do**
  If $(\boldsymbol{\mu}_i > 0)$
  $\mathbf{X1}(\mathbf{j}, :) = \mathbf{X}(\mathbf{i}, :)$ and $\mathbf{Y1}(\mathbf{j}, \mathbf{1}) = \mathbf{Y}(\mathbf{i}, \mathbf{1})$
  $j \leftarrow j + 1$
end for
for $i = 1 \rightarrow N$ **do**
  If $(\boldsymbol{\mu}_i = 0)$
  $\mathbf{X2}(\mathbf{k}, :) = \mathbf{X}(\mathbf{i}, :)$ and $\mathbf{Y2}(\mathbf{k}, \mathbf{1}) = \mathbf{Y}(\mathbf{i}, \mathbf{1})$
  $k \leftarrow k + 1$
end for

6: Use classifier to predict the final class of $\mathbf{X2}$.

$N_e << N$. Therefore, $\mathbf{X_{truncated}}$ and $\Lambda_{\mathbf{truncated}}$ are matrices of sizes $N \times N_e$ and $N_e \times N_e$; thus, at least one dimension is very small. One can then rewrite (19) as

$$\mathbf{U^{n+\frac{1}{2}}} = \mathbf{X_{truncated}}(\mathbf{I} + dt\,\Lambda_{\mathbf{truncated}})^{-1}\mathbf{X^{T}_{truncated}}\mathbf{U_{update}}, \tag{20}$$

where $\mathbf{U_{update}} = \mathbf{U^{n}} - dt\boldsymbol{\mu} \cdot (\mathbf{U}^n - \mathbf{U_{labeled}})$. Note that all of the aforementioned matrices in (20) have at least one dimension which is small, and that only the smallest N_e eigenvalues of the persistent Laplacian and their corresponding eigenvectors need to be computed, saving much computational time.

Overall, this spectral technique allows the diffusion operation of the proposed scheme, particularly Step 1, which involves the heat equation with a forcing term, to be decomposed into faster matrix multiplication, resulting in an efficient proposed algorithm. It is also important to note that the graph weights are only used to compute the first N_e eigenvalues. Once they are computed, the main steps of the scheme involve only the truncated matrices and $\mathbf{U}$, which allows the scheme to be fast.

Regarding the computation of the few eigenvalues and their corresponding eigenvectors (for a particular persistent Laplacian derived in Sect. 2.3), which our proposed method requires, we note that there are many methods available for the task. For sparse matrices, such as the family of persistent Laplacians derived in this paper, and moderately small datasets, the authors in Bertozzi and Flenner (2012), Garcia-Cardona et al. (2014) suggest using the Rayleigh–Chebyshev procedure (Anderson, 2010). This efficient method is a modified version of the inverse subspace iteration algorithm. For large and fully connected graphs, the Nyström extension technique (Belongie et al., 2002; Fowlkes et al., 2001) is recommended. In particular, the Nyström extension algorithm is a matrix completion method that incorporates computations using much smaller submatrices of lower dimensions, thus saving computational time, and approximates eigenvectors and eigenvalues using a quadrature rule with randomly chosen interpolation points. Moreover, this technique requires the computation of only a very small portion of the graph weights, making this procedure efficient even for very large datasets. For simplicity, in our computational experiments, we use MATLAB's *eigs* function to compute the N_e smallest eigenvalues and the associated eigenvectors. The details on MATLAB's *eigs* function is provided in supplementary material.

For small data sets, like some of those used in the experiments of this paper, it is more desirable to compute the graph weights directly by calculating pairwise distances. In this case, the efficiency of the task can be increased by using a parallel computing technique or by reducing the dimension of the data. Then, a graph is often made sparse using, for example, thresholding or an l nearest neighbors technique, resulting in a similarity graph where most of the edge weights are zero. Overall, a nearest neighbor graph can be computed efficiently using, for example, the kd-tree code of the VLFeat open source library (Vedaldi and Fulkerson, 2008).

4 Results and discussion

4.1 Data sets

We tested our proposed method on five benchmark data sets: two artificial data sets and five real-world data sets. The data sets are as follows:

- The G50C data set (G50C, 2009) is an artificial data set inspired by Grandvalet and Bengio (2004): the data is generated from two standard normal multivariate Gaussians. This data set contains 550 points located in a 50-dimensional space such that the Bayes error is 5%. There are two classes in the data set.
- The WebKB, i.e., World Wide Knowledge Base, data set (WebKB, 1998) contains web pages collected from departments of Cornell University, University of Texas, University of Washington, and University of Wisconsin. We used a subset of the WebKB data set consisting of 1051 sample points that were classified into two categories: course and non-course. Each web document is described by the text on the web pages (called page representation) and the anchor text on the hyperlinks pointing to the page (called link representation). The information from the text of each data element is encoded with a 4840-component vector.
- The Pendigits data set (Pendigits, 1998) is a set of 10,992 images of handwritten digits; the set has 10 classes. Each image is represented by a feature vector consisting of 16 values, each between 0 and 100.
- The Statlog Heart data set (Heart, 1998) is a data set of 270 elements with 13 attributes. There are 2 classes, predicting the absence or presence of heart disease.
- The Madelon data set (Madelon, 2008) consists of data points grouped into 32 clusters, placed at the vertices of a five-dimensional hypercube, with two classes. The five dimensions constitute five informative features, and 15 linear combinations of those features were added to form a set of 20 redundant informative features. The goal is to classify examples into two classes based on these 20 features. The dataset also contains distractor features called 'probes' with no predictive power. The order of the features and patterns was randomized. This data set contains 2000 elements, all of which are in a 500-dimensional space.
- The Banana data set (Banana, 2015) is a binary classification dataset that contains two banana-shaped clusters; thus, there are two classes in this data set. There are 5300 elements in the data set, which each element being 2-dimensional.
- The Opt-Digits data set (OptDigits, 1998) is a multiclass data set of grayscale images of 5620 handwritten digits. It has 5620 instances with 64 attributes. The data set has 10 classes.
- The USPS data set (USPS, 2015) is a multiclass data set of 9298 grayscale images with 10 different classes. The grayscale images are centered, mormalized and show a broad range of font styles.
- The Coil-20 data set (COIL-20, 1996) is a multiclass data set of 1440 normalized images of 20 objects. The objects were placed on motornized turn table. With fixed camera, the turn table was rotated trough 360 degrees to capture different pose on the interval of 5 degrees.
- The Landsat data set (Landsat, 1999) is a set of 6435 elements which consist of the multi-spectral values of pixels of 3×3 neighbourhoods in a satellite image, and the classification is associated with the central pixel in each neighbourhood.
- The CIFAR-10 data set (Krizhevsky et al., 2009) is a set of 60,000, 32×32 colour images of 10 classes. Each class has 6000 images. To build good quality graphs, we trained autoencoders to extract important features from the data. We used AutoEncodingTransformations architecture (Zhang et al., 2019) with default paramaters for the training.
- The CIFAR-100 data set (Krizhevsky et al., 2009) is a set of 60,000, 32×32 colour images of 100 classes. Each class has 600 images. To build good quality graphs, we trained variational autoencoders (Kingma and Welling, 2013) to extract representation

Table 1 Data sets used in the experiments

Data set	Number of data elements	Number of attributes	Number of classes
G50C	550	50	2
Opt-Digits	5620	64	10
Heart	270	13	2
WebKB	1051	4840	2
Madelon	2000	500	2
Banana	5300	2	2
Coil-20	1440	1024	20
USPS	9298	256	10
Landsat	6435	36	6
Pendigits	10,992	16	10
CIFAR-10	60,000	3072	10
CIFAR-100	60,000	3072	100

of the data in the latent space. We run 200 epochs with the default parameters for the training.

The details of the data sets are outlined in Table 1.

4.2 Hyperparameters selection

In this section, we outline the parameters that we have selected for the proposed method. Instead of computing the full graph, we construct an N_n-nearest neighbor graph. To compute the graph weights, we use the Zelnik-Manor and Perona (ZMP) weight function (2). After the graph construction, we compute the family of persistent graph Laplacians as derived in Sect. 2.3.4. For each persistent Laplacian in the family, we compute its first N_e eigenvalues and eigenvectors, where $N_e << N$. Overall, both N_n and N_e are hyperparameters that need tuning. Some other hyperparameters that might require tuning are the number of persistent Laplacians (L_n), the time step for solving the heat equation (dt), the constraint constant on the fidelity term (μ), the maximum number of iterations (N_t), and the factor C in the diffusion operator. In this paper, we found the exact parameters used for the experiments using random search and outlined them in the Supplementary Information.

4.3 Comparison to recent methods

In this section, we compare our proposed method to several recent graph-based semi-supervised algorithms. The results of the experiments and the comparison methods are shown in Table 2.Please check the layout of Tables 2 and 3 are correct. All computational experiments were implemented using the GraphLearning Python package (Calder, 2022).

In particular, for all data sets, we compare our proposed method, PL-MBO, to the following recent graph based semi-supervised algorithms:

- centered kernel method (CK) (Mai and Couillet, 2018)
- p-Laplace method (p-Laplace) (Flores et al., 2022)
- modularity MBO (M-MBO) (Boyd et al., 2018)

- SSL via absolutely minimal Lipschitz extension ((AMLE) (Bungert et al., 2023)
- Poisson MBO (P-MBO) (Calder et al., 2020)

Below, we provide a brief overview of the previously mentioned semi-supervised methods. In particular, in Mai and Couillet (2018), the authors generalize the graph-based semi-supervised technique proposed in Zhu et al. (2003), which derives an approach to semi-supervised learning based on a Gaussian random field model. Specifically, they introduce a normalization parameter in the cost function in order to construct a large class of regularized affinity-based methods, among which are the Laplacian-based techniques. They then provide a quantitative performance study of the generalized graph-based semi-supervised method for large dimensional Gaussian-mixture data and radial kernels. In Flores et al. (2022), the authors explore the graph p-Laplacian, where $p > 2$, as a replacement for the standard ($p = 2$) graph Laplacian, for graph-based semi-supervised learning in the case of low amounts of labeled data. In addition, they develop fast and scalable procedures for solving the variational and game-theoretic p-Laplace equations on weighted graphs for $p > 2$. Overall, Flores et al. (2022) derives the theory and explores applications of l_p-based Laplacian regularization in semi-supervised learning. In Boyd et al. (2018), the authors propose a modularity optimization scheme to perform semi-supervised learning on graphs. In Calder and Ettehad (2022), the authors study a family of Hamilton-Jacobi equations on graphs; the equations are named p-eikonal equations. The authors also consider the application of p-eikonal equations on semi-supervised learning. The experiments on real image datasets shows that p-eikonal equations offers significantly better results compared to those of the shortest part distances metric. In Bungert et al. (2023), the authors prove uniform convergence rates for solutions of the graph infinite Laplace equation as the number of vertices grows to infinity. In Calder et al. (2020), the authors propose a Poisson learning framework for graph based semi-supervised learning at a very low label rates. One key difference between Poisson learning and the classical Laplace learning framework is that in Laplace learning, the labels are imposed as boundary conditions in a Laplace equation, while in Poisson learning, the labels appear as a source term in the graph Poisson equation.

4.4 Performance and discussion

The data sets that we used for our computational experiments are detailed in Sect. 4.1 and Table 1. For all data sets, we consider the AUC score as the main evaluation metric. The results of the experiments are shown in Table 2. In terms of the AUC score, our proposed method obtains superior results on all data sets.

4.5 Statistical tests

We performed a non-parametric test regarding the experiments obtained from the learning algorithms, specifically, the Friedman ranking test, which assigns rankings to each data set. The statistic follows a chi-squared distribution with $K - 1$ degrees of freedom, where K represents the number of algorithms. Moreover, the null hypothesis (H0) for the Friedman test is that there are no differences between the methods. In this case, the null hypothesis was rejected due to very low p-value. The results from the test are in Table 3, where we also record the mean rank value of the Friedman test for each algorithms, where smaller ranks indicate a more accurate algorithm. The proposed PL-MBO algorithm obtained a superior ranking.

Table 2 AUC score comparison of the PL-MBO algorithm with other methods

# of labels per class	5	10	15	20	25
(a) G50C results (AUC score)					
PL-MBO	**0.978**	**0.983**	**0.986**	**0.986**	**0.986**
CK	0.958	0.959	0.960	0.961	0.961
p-Laplace	0.955	0.981	0.983	0.985	0.985
M-MBO	0.954	0.954	0.955	0.957	0.957
AMLE	0.938	0.940	0.943	0.943	0.954
P-MBO	0.934	0.935	0.942	0.945	0.948
(b) Opt-Digits results (AUC score)					
PL-MBO	**0.992**	**0.998**	**0.998**	**0.998**	**0.998**
CK	0.953	0.969	0.974	0.979	0.981
p-Laplace	0.993	0.995	0.995	0.996	0.997
M-MBO	0.981	0.986	0.986	0.988	0.987
AMLE	0.994	0.994	0.996	0.997	0.997
P-MBO	0.981	0.986	0.987	0.988	0.988
(c) Heart results (AUC score)					
PL-MBO	**0.836**	**0.870**	**0.868**	**0.871**	0.873
CK	0.784	0.797	0.807	0.814	0.826
p-Laplace	0.783	0.819	0.830	0.862	**0.882**
M-MBO	0.791	0.793	0.802	0.797	0.830
AMLE	0.759	0.736	0.740	0.757	0.764
P-MBO	0.801	0.812	0.828	0.834	0.846
(d) WebKB results (AUC score)					
PL-MBO	**0.974**	**0.984**	0.983	0.985	0.985
CK	0.969	0.969	0.970	0.970	0.971
p-Laplace	0.972	0.982	**0.985**	**0.994**	**0.995**
M-MBO	0.954	0.958	0.959	0.962	0.962
AMLE	0.949	0.942	0.924	0.930	0.946
P-MBO	0.922	0.916	0.902	0.905	0.905
(e) Madelon results (AUC score)					
PL-MBO	0.587	0.604	**0.655**	0.657	**0.678**
CK	0.568	0.583	0.595	0.601	0.615
p-Laplace	0.588	**0.613**	0.640	**0.658**	0.671
M-MBO	0.541	0.580	0.607	0.589	0.594
AMLE	**0.591**	0.596	0.617	0.634	0.638
P-MBO	0.567	0.586	0.608	0.614	0.625
(f) Banana results (AUC score)					
PL-MBO	**0.760**	0.798	0.859	**0.890**	**0.903**
CK	0.689	0.751	0.786	0.804	0.826
p-Laplace	0.727	**0.818**	0.826	0.872	0.895
M-MBO	0.747	0.779	0.781	0.811	0.804
AMLE	0.758	0.828	**0.860**	0.888	0.898
P-MBO	0.688	0.764	0.803	0.813	0.834

The best results among all methods are indicated in bold

Table 2 continued

# of labels per class	5	10	15	20	25
(g) Coil20 results (AUC score)					
PL-MBO	**0.993**	**0.996**	**0.997**	**0.998**	**0.998**
CK	0.924	0.961	0.975	0.984	0.993
p-Laplace	0.990	0.994	0.995	0.996	0.997
M-MBO	0.747	0.779	0.781	0.811	0.804
AMLE	0.988	0.993	0.995	0.996	0.997
P-MBO	0.959	0.976	0.985	0.993	0.995
(h) USPS results (AUC score)					
PL-MBO	**0.991**	**0.993**	**0.993**	**0.993**	**0.993**
CK	0.906	0.930	0.940	0.947	0.953
p-Laplace	0.972	0.980	0.982	0.985	0.986
M-MBO	0.953	0.960	0.964	0.964	0.964
AMLE	0.983	0.980	0.984	0.987	0.990
P-MBO	0.952	0.959	0.961	0.965	0.967
(i) Landsat results (AUC score)					
PL-MBO	**0.959**	**0.971**	**0.975**	**0.977**	**0.978**
CK	0.835	0.900	0.918	0.937	0.945
p-Laplace	0.950	0.960	0.962	0.965	0.967
M-MBO	0.876	0.892	0.897	0.901	0.901
AMLE	0.950	0.955	0.961	0.966	0.971
P-MBO	0.886	0.898	0.904	0.908	0.910
(j) Pendigits results (AUC score)					
PL-MBO	**0.981**	**0.988**	**0.990**	**0.992**	**0.993**
CK	0.912	0.949	0.961	0.969	0.972
p-Laplace	0.956	0.972	0.980	0.984	0.986
M-MBO	0.957	0.973	0.977	0.982	0.986
AMLE	0.936	0.953	0.959	0.970	0.968
P-MBO	0.961	0.970	0.976	0.983	0.985
(k) CIFAR-10 results (AUC score)					
PL-MBO	**0.894**	**0.912**	**0.917**	**0.920**	**0.928**
CK	0.801	0.831	0.847	0.855	0.858
p-Laplace	0.847	0.871	0.879	0.883	0.886
M-MBO	0.735	0.760	0.767	0.768	0.773
AMLE	0.853	0.871	0.873	0.879	0.884
P-MBO	0.764	0.790	0.796	0.802	0.809
(l) CIFAR-100 results (AUC score)					
PL-MBO	**0.751**	**0.778**	**0.801**	**0.810**	**0.823**
CK	0.641	0.679	0.695	0.717	0.734
p-Laplace	0.684	0.711	0.726	0.738	0.744
M-MBO	0.583	0.621	0.643	0.667	0.689
AMLE	0.676	0.695	0.721	0.723	0.731
P-MBO	0.695	0.733	0.754	0.767	0.779

The best results among all methods are indicated in bold

Table 3 Friedman test (significance level of 0.05)

Statistic	p-value	Result
24.72579	0.00016	H0 is rejected
Rank	**Ranking algorithms**	
19.36364	PL-MBO	
22.45455	p-Laplace	
27.72727	AMLE	
42.00000	CK	
42.27273	P-MBO	
47.18182	M-MBO	

Table 4 Ablation studies

Data set	Accuracy (with combinatorial Laplacian)	Accuracy (with the proposed persistent Laplacian)
G50C	93.08	95.18
Opt-Digits	96.95	97.85
Heart	80.09	81.00
WebKB	97.37	97.63
Madelon	60.10	62.34
Banana	83.47	85.05
Coil-20	77.35	89.16
USPS	91.48	94.85
Landsat	82.29	83.73
Pendigits	87.55	93.22
CIFAR-10	0.897 (AUC)	0.928 (AUC)
CIFAR-100	0.780 (AUC)	0.823 (AUC)

4.6 Ablation studies

In order to evaluate the effectiveness of our method, we conducted ablation studies on various components of our method. In particular, we analyzed the effectiveness of using the proposed persistent Laplacian instead of using the standard combinatorial Laplacian. The results are shown in Table 4. It is important to note that the use of the persistent Laplacian has improved the accuracy on all datasets.

4.7 Timing

The proposed method is very efficient; all experiments were performed on a 1.4 GHz Quad-Core Interl Core i5 computer. The timing results are listed in Table 5, where we record the timing details of each data set. In particular, we divide the timing into two parts. First, we record the time required to compute the graph weights. Second, we record the time required for the PL-MBO procedure once the graph has been computed. We perform the experiment ten times and record the average value.

Table 5 The timing of the proposed PL-MBO method

Data set	Number of data elements	Number of attributes	Timing (construction of weight matrix) (s)	Timing (PL-MBO procedure) (s)
G50C	550	50	0.02	0.25
Heart	270	13	0.004	0.30
WebKB	1051	4840	0.03	0.86
Coil-20	1440	1024	0.06	8.45
Madelon	2000	500	0.12	14.59
Banana	5300	2	1.23	55.84
Opt-Digits	5620	64	1.60	139.61
Landsat	6435	36	1.95	157.55
USPS	9298	256	11.03	217.81
Pendigits	10,992	16	13.90	421.12
CIFAR-10	60,000	3072	23.64*	3053.23*
CIFAR-100	60,000	3072	25.72*	3956.68*

*These calculations were made on a MacBook Pro (Apple M3 Max) with 64GB RAM memory. We used 20-dimensional autoencoder-based embeddings for CIFAR-10 and CIFAR-100

5 Concluding remarks

We present a novel topological graph-based semi-supervised method called PL-MBO by integrating persistent spectral graphs with an adaptation and graph-based modification of the classical Merriman–Bence–Osher (MBO) technique. This method is an efficient procedure that performs well with low label rates and small amounts of labeled data, which is crucial since labeled data is often scarce in many applications. The proposed algorithm is also adaptable for both small and large datasets. Experimental results on various benchmark datasets indicate that the proposed PL-MBO method outperforms other recent methods. In future work, we plan to explore integrating techniques such as various types of autoencoders for feature extraction, and to perform experiments with different types of molecular data. Overall, the proposed PL-MBO scheme is a powerful approach for data science when there are few labeled data elements, a common scenario for many applications.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10994-024-06616-w.

Author Contributions G.B. wrote the main manuscript text and conducted the majority of the computational experiments, while E.M. and G.W. edited the resulting manuscript, and conducted some of the computational experiments.

Funding This work was supported in part by NSF Grant DMS-2052983 and NIH Grant R01AI164266.

Code availability The source code is available on Github at https://github.com/kmerkurev/Persistent-Laplacian-Method.

Declarations

Conflict of interest The authors do not have any conflict of interest.

Ethical approval This paper does not require special ethics approval

References

Ahmed, A., Shervashidze, N., Narayanamurthy, S., Josifovski, V., & Smola, A. J. (2013). Distributed large-scale natural graph factorization. In *Proceedings of the 22nd international conference on World Wide Web* (pp. 37–48).

Anderson, C. R. (2010). A Rayleigh–Chebyshev procedure for finding the smallest eigenvalues and associated eigenvectors of large sparse Hermitian matrices. *Journal of Computational Physics, 229*(19), 7477–7487.

Banana. (2015). *Banana Data Set*. https://sci2s.ugr.es/keel/category.php?cat=clas

Belkin, M., Matveeva, I., & Niyogi, P. (2004a). Regularization and semi-supervised learning on large graphs. In *17th Annual conference on learning theory* (pp. 624–638).

Belkin, M., Matveeva, I., & Niyogi, P. (2004). Tikhonov regularization and semi-supervised learning on large graphs. *In IEEE international conference on acoustics, speech, and signal processing, 3*, iii–1000.

Belkin, M., & Niyogi, P. (2001). Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in Neural Information Processing Systems, 14*, 66.

Belkin, M., Niyogi, P., & Sindhwani, V. (2006). Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research, 7*(11), 66.

Belongie, S., Fowlkes, C., Chung, F., & Malik, J. (2002). Spectral partitioning with indefinite kernels using the Nyström extension. In *7th European conference on computer vision* (pp. 531–542).

Bertozzi, A. L., & Flenner, A. (2012). Diffuse interface models on graphs for classification of high dimensional data. *Multiscale Modeling & Simulation, 10*(3), 1090–1118.

Boyd, Z. M., Bae, E., Tai, X.-C., & Bertozzi, A. L. (2018). Simplified energy landscape for modularity using total variation. *SIAM Journal on Applied Mathematics, 78*(5), 2439–2464.

Bungert, L., Calder, J., & Roith, T. (2023). Uniform convergence rates for Lipschitz learning on graphs. *IMA Journal of Numerical Analysis, 43*(4), 2445–2495.

Cai, Y., Zhang, Z., Cai, Z., Liu, X., Ding, Y., & Ghamisi, P. (2021). *Fully linear graph convolutional networks for semi-supervised learning and clustering*. arXiv preprint arXiv:2111.07942

Calder, J. (2022). *Graph learning python package.* https://doi.org/10.5281/zenodo.5850940

Calder, J., Brendan, C., Thorpe, M., & Slepcev, D. (2020). Poisson learning: Graph based semi-supervised learning at very low label rates. In *International conference on machine learning* (pp. 1306–1316). PMLR.

Calder, J., & Ettehad, M. (2022). Hamilton–Jacobi equations on graphs with applications to semi-supervised learning and data depth. *The Journal of Machine Learning Research, 23*(1), 14267–14328.

Cao, S., Lu, W., & Xu, Q. (2015). GraRep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM international on conference on information and knowledge management* (pp. 891–900).

Cao, S., Lu, W., & Xu, Q. (2016). Deep neural networks for learning graph representations. In *Proceedings of the AAAI conference on artificial intelligence* (vol. 30).

Chen, H., Perozzi, B., Hu, Y., & Skiena, S. (2018). HARP: Hierarchical representation learning for networks. In *Proceedings of the AAAI conference on artificial intelligence* (vol. 32).

Chen, J., Qiu, Y., Wang, R., & Wei, G.-W. (2022). Persistent Laplacian projected Omicron BA. 4 and BA. 5 to become new dominating variants. *Computers in Biology and Medicine, 151*, 106262.

Chen, Y., & Ye, X. (2011). *Projection onto a simplex*. arXiv preprint arXiv:1101.6081

COIL-20. (1996). *COIL-20 Data Set*. Technical report CUCS-005-96.

Edelsbrunner, H., & Harer, J. (2008). Persistent homology—A survey. *Contemporary Mathematics, 453*(26), 257–282.

Flores, M., Calder, J., & Lerman, G. (2022). Analysis and algorithms for ℓ_p-based semi-supervised learning on graphs. *Applied and Computational Harmonic Analysis, 60*, 77–122.

Fowlkes, C., Belongie, S., & Jitendra, M. (2001). Efficient spatiotemporal grouping using the Nyström method. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition* (vol. 1, pp. I-I).

Fu, S., Liu, W., Guan, W., Zhou, Y., Tao, D., & Xu, C. (2021a). Dynamic graph learning convolutional networks for semi-supervised classification. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 17*(1s), 1–13.

Fu, S., Liu, W., Zhang, K., Zhou, Y., & Tao, D. (2021b). Semi-supervised classification by graph p-Laplacian convolutional networks. *Information Sciences, 560*, 92–106.

G50C. (2009). *G50C Data Set*. http://vikas.sindhwani.org/datasets/ssl/

Garcia-Cardona, C., Merkurjev, E., Bertozzi, A. L., Flenner, A., & Percus, A. G. (2014). Multiclass data segmentation using diffuse interface methods on graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 36*(8), 1600–1613.

Gong, C., Liu, T., Tao, D., Keren, F., Enmei, T., & Yang, J. (2015). Deformed graph Laplacian for semi-supervised learning. *IEEE Transactions on Neural Networks and Learning Systems, 26*(10), 2261–2274.
Grandvalet, Y., & Bengio, Y. (2004). Semi-supervised learning by entropy minimization. *Advances in Neural Information Processing Systems, 17*, 66.
Grover, A., & Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 855–864).
Hatcher, A. (2005). *Algebraic topology*.
Hayes, N., Merkurjev, E., & Wei, G.-W. (2023). Integrating transformer and autoencoder techniques with spectral graph algorithms for the prediction of scarcely labeled molecular data. *Computers in Biology and Medicine, 153*, 106479.
Heart. (1998). *Statlog heart data set*.
Jacobs, M., Merkurjev, E., & Esedo$\bar{g}$lu, S. (2018). Auction dynamics: A volume constrained mbo scheme. *Journal of Computational Physics, 354*, 288–310.
Jiang, B., & Lin, D. (2018). *Graph Laplacian regularized graph convolutional networks for semi-supervised learning*. arXiv preprint arXiv:1809.09839
Jung, A., Hero, A. O., III, Mara, A., & Jahromi, S. (2016). *Semi-supervised learning via sparse label propagation*. arXiv preprint arXiv:1612.01414
Kingma, D. P., & Welling, M. (2013). *Auto-encoding variational Bayes*. arXiv preprint arXiv:1312.6114
Kipf, T., & Max, W. (2017). Semi-supervised classification with graph convolutional networks. In *Proceedings of 5th international conference on learning representations* (pp. 1–14).
Krizhevsky, A., & Hinton, G. (2009). *Learning multiple layers of features from tiny images*.
Landsat. (1999). *Landsat Data Set*. https://archive.ics.uci.edu/dataset/146/statlog+landsat+satellite
Li, R., Wang, S., Zhu, F., & Huang, J. (2018). Adaptive graph convolutional neural networks. In *Proceedings of the AAAI conference on artificial intelligence* (vol. 32).
Liu, W., He, J., & Chang, S.-F. (2010). Large graph construction for scalable semi-supervised learning. In *Proceedings of the 27th international conference on machine learning* (pp. 679–686).
Madelon. (2008). *Madelon Data Set*. https://archive.ics.uci.edu/ml/machine-learning-databases/madelon/
Mai, X., & Couillet, R. (2018). A random matrix analysis and improvement of semi-supervised learning for large dimensional data. *The Journal of Machine Learning Research, 19*(1), 3074–3100.
Mémoli, F., Wan, Z., & Wang, Y. (2022). Persistent Laplacians: Properties, algorithms and implications. *SIAM Journal on Mathematics of Data Science, 4*(2), 858–884.
Meng, Z., Merkurjev, E., Koniges, A., & Bertozzi, A. L. (2017). Hyperspectral image classification using graph clustering methods. *Image Processing On Line, 7*, 218–245.
Meng, Z., & Xia, K. (2021). Persistent spectral-based machine learning (PerSpect ML) for protein–ligand binding affinity prediction. *Science Advances, 7*(19), eabc5329.
Merkurjev, E. (2020). A fast graph-based data classification method with applications to 3D sensory data in the form of point clouds. *Pattern Recognition Letters, 136*, 154–160.
Merkurjev, E., Bertozzi, A. L., & Chung, F. (2018). A semi-supervised heat kernel pagerank mbo algorithm for data classification. *Communications in Mathematical Sciences, 16*(5), 1241–1265.
Merkurjev, E., Garcia-Cardona, C., Bertozzi, A. L., Flenner, A., & Percus, A. G. (2014). Diffuse interface methods for multiclass segmentation of high-dimensional data. *Applied Mathematics Letters, 33*, 29–34.
Merkurjev, E., Kostic, T., & Bertozzi, A. L. (2013). An MBO scheme on graphs for classification and image processing. *SIAM Journal on Imaging Sciences, 6*(4), 1903–1930.
Merkurjev, E., Nguyen, D. D., & Wei, G.-W. (2022). Multiscale Laplacian learning. *Applied Intelligence, 66*, 1–20.
Merkurjev, E., Sunu, J., & Bertozzi, A. L. (2014b). Graph MBO method for multiclass segmentation of hyperspectral stand-off detection video. In *IEEE international conference on image processing* (pp. 689–693).
Merriman, B., Bence, J. K., & Osher, S. J. (1994). Motion of multiple junctions: A level set approach. *Journal of Computational Physics, 112*(2), 334–363.
Nie, F., Xiang, S., Liu, Y., & Zhang, C. (2010). A general graph-based semi-supervised learning with novel class discovery. *Neural Computing and Applications, 19*, 549–555.
OptDigits. (1998). Optical recognition of handwritten digits. *UCI Machine Learning Repository*. https://doi.org/10.24432/C50P49
Ou, M., Cui, P., Pei, J., Zhang, Z., & Zhu, W. (2016). Asymmetric transitivity preserving graph embedding. In *ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1105–1114).
Pendigits. (1998). *Pen-based recognition of handwritten digits data set*. https://archive.ics.uci.edu/ml/datasets/Pen-Based+Recognition+of+Handwritten+Digits
Perozzi, B., Al-Rfou, R., & Skiena, S. (2014). Deepwalk: Online learning of social representations. In *ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 701–710).

Roweis, S. T., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science, 290*(5500), 2323–2326.
Singhal, A. (2001). Modern information retrieval: A brief overview. *IEEE Data Engineering Bulletin, 24*(4), 35–43.
Song, Z., Yang, X., Zenglin, X., & King, I. (2022). Graph-based semi-supervised learning: A comprehensive review. *IEEE Transactions on Neural Networks and Learning Systems. 6*, 66.
Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., & Mei, Q. (2015). LINE: Large-scale information network embedding. In *Proceedings of the international conference on World Wide Web* (pp. 1067–1077).
Tu, K., Cui, P., Wang, X., Yu, P. S., & Zhu, W. (2018). Deep recursive network embedding with regular equivalence. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 2357–2366).
USPS. (2015). *Usps data set*. https://www.kaggle.com/datasets/bistaumanga/usps-dataset
Van Engelen, J. E., & Hoos, H. H. (2020). A survey on semi-supervised learning. *Machine Learning, 109*(2), 373–440.
Vedaldi, A., & Fulkerson, B. (2008). *VLFeat: An open and portable library of computer vision algorithms*. http://www.vlfeat.org/
Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2018). Graph attention networks. In *Proceedings of the international conference on learning representations,* (vol. 1050(no. 20), pp. 1–12).
Wang, B., Tu, Z., Tsotsos, J. K. (2013). Dynamic label propagation for semi-supervised multi-class multi-label classification. In *Proceedings of the IEEE international conference on computer vision* (pp. 425–432).
Wang, D., Cui, P. & Zhu, W. (2016). Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1225–1234).
Wang, F., & Zhang, C. (2006). Label propagation through linear neighborhoods. In *Proceedings of the 23rd international conference on machine learning* (pp. 985–992).
Wang, F., Zhu, L., Xie, L., Zhang, Z., & Zhong, M. (2021). Label propagation with structured graph learning for semi-supervised dimension reduction. *Knowledge-Based Systems, 225*, 107130.
Wang, R., Nguyen, D. D., & Wei, G.-W. (2020). Persistent spectral graph. *International Journal for Numerical Methods in Biomedical Engineering, 36*(9), e3376.
Wang, X., Ji, H., Shi, C., Wang, B., Ye, Y., Cui, P., & Yu, P. S. (2019). Heterogeneous graph attention network. In *The World Wide Web conference* (pp. 2022–2032).
WebKB. (1998). *vCMU World Wide Knowledge Base (WebKB) project*. http://www.cs.cmu.edu/~webkb/
Xu, B., Shen, H., Cao, Q., Cen, K., & Cheng, X. (2020). *Graph convolutional networks using heat kernel for semi-supervised learning*. arXiv preprint arXiv:2007.16002
Xu, Z., King, I., Lyu, M.R.-T., & Jin, R. (2010). Discriminative semi-supervised feature selection via manifold regularization. *IEEE Transactions on Neural Networks, 21*(7), 1033–1047.
Yang, Z., Cohen, W., & Salakhudinov, R. (2016). Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning* (pp. 40–48).
Zelnik-Manor, L., & Perona, P. (2004). Self-tuning spectral clustering. *Advances in Neural Information Processing Systems, 17*, 66.
Zhang, J., Shi, X., Xie, H. M., Junyuan, A., King, I., & Yeung, D.-Y. (2018). GaAN: Gated attention networks for learning on large and spatiotemporal graphs. In *Proceedings of the thirty-fourth conference on uncertainty in artificial intelligence* (pp. 339–349).
Zhang, L., Qi, G.-J., Wang, L., & Luo, J. (2019). Aet vs. aed: Unsupervised representation learning by auto-encoding transformations rather than data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 2547–2555).
Zhou, D., Bousquet, O., Lal, T., Weston, J., & Schölkopf, B. (2003). Learning with local and global consistency. *Advances in Neural Information Processing Systems, 16*, 66.
Zhou, D., Huang, J., & Schölkopf, B. (2005). Learning from labeled and unlabeled data on a directed graph. In *Proceedings of international conference on machine learning* (pp. 1036–1043).
Zhou, X., Liu, X., Yu, H., Wang, J., Xie, Z., Jiang, J., & Ji, X. (2023). Variance-enlarged poisson learning for graph-based semi-supervised learning with extremely sparse labeled data. In *The twelfth international conference on learning representations*.
Zhu, X., & Ghahramani, Z. (2002). *Learning from labeled and unlabeled data with label propagation*. CMU CALD Tech Report CMU-CALD-02-107.
Zhu, X., Ghahramani, Z., & Lafferty, J. (2003). Semi-supervised learning using Gaussian fields and harmonic functions. *International Conference on Machine Learning, 3*, 912.
Zomorodian, A., & Carlsson, G. (2004). Computing persistent homology. In *Proceedings of the twentieth annual symposium on computational geometry* (pp. 347–356).