Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 1619 – 1635
© TÜBİTAK
doi:10.3906/elk-1812-124

Domain adaptation on graphs by learning graph topologies: theoretical analysis and an algorithm

Elif VURAL
Department of Electrical and Electronics Engineering, Middle East Technical University, Ankara, Turkey

Received: 18.12.2018 • **Accepted/Published Online:** 06.03.2019 • **Final Version:** 15.05.2019

Abstract: Traditional machine learning algorithms assume that the training and test data have the same distribution, while this assumption does not necessarily hold in real applications. Domain adaptation methods take into account the deviations in data distribution. In this work, we study the problem of domain adaptation on graphs. We consider a source graph and a target graph constructed with samples drawn from data manifolds. We study the problem of estimating the unknown class labels on the target graph using the label information on the source graph and the similarity between the two graphs. We particularly focus on a setting where the target label function is learned such that its spectrum is similar to that of the source label function. We first propose a theoretical analysis of domain adaptation on graphs and present performance bounds that characterize the target classification error in terms of the properties of the graphs and the data manifolds. We show that the classification performance improves as the topologies of the graphs get more balanced, i.e. as the numbers of neighbors of different graph nodes become more proportionate, and weak edges with small weights are avoided. Our results also suggest that graph edges between too distant data samples should be avoided for good generalization performance. We then propose a graph domain adaptation algorithm inspired by our theoretical findings, which estimates the label functions while learning the source and target graph topologies at the same time. The joint graph learning and label estimation problem is formulated through an objective function relying on our performance bounds, which is minimized with an alternating optimization scheme. Experiments on synthetic and real data sets suggest that the proposed method outperforms baseline approaches.

Key words: Domain adaptation, data classification, graph Fourier basis, graph Laplacian, performance bounds

1. Introduction

Classical machine learning methods are based on the assumption that the training data and the test data have the same distribution. A classifier is trained on the training data, which is then used to estimate the unknown class labels of the test data. On the other hand, domain adaptation methods consider a setting where the distribution of the test data is different from that of the training data [1–4]. Given many labeled samples in a source domain and much fewer labeled samples in a target domain, domain adaptation algorithms exploit the information available in both domains in order to improve the performance of classification in the target domain. Meanwhile, many machine learning applications nowadays involve inference problems on graph domains such as social networks and communication networks. Moreover, in most problems data samples conform to a low-dimensional manifold model; for instance, face images of a person captured from different camera angles lie on a low-dimensional manifold. In such problems, graphs models are widely used as they are very convenient for

*Correspondence: velif@metu.edu.tr

This work has been supported by the TÜBİTAK 2232 program, project no. 117C007.

approximating the actual data manifolds. Hence, domain adaptation on graphs arises as an important problem of interest, which we study in this work. We first present performance bounds for transferring the knowledge of class labels between a pair of graphs. We then use these bounds to develop an algorithm that computes the structures of the source and the target graphs and estimates the unknown class labels at the same time.

Many graph-based learning methods rely on the assumption that the label function varies slowly on the data graph. However, this assumption does not always hold as shown in Figure 1(a). A face manifold is illustrated in Figure 1(a), each point of which corresponds to a face image that belongs to one of three different subjects. In this example, the class label function varies slowly along the blue direction, where images belong to the same subject. On the other hand, along the red direction the face images of different subjects get too close to each other due to extreme illumination conditions. Hence, the label function has fast variation along the red direction on the manifold.

Although it is common to assume that the label function varies slowly in problems concerning a single graph, in a problem with more than one graph it is possible to learn the speed of variation of the label function and share this information across different graphs. This is illustrated in Figure 1(b), where the characteristics of the variation of the label function can be learnt on a source graph where many class labels are available. Then, the purpose of graph domain adaptation is to transfer this information to a target graph that contains very few labels and estimate the unknown class labels. We studied this problem in our previous work [5], where we proposed a method called spectral domain adaptation (SDA).

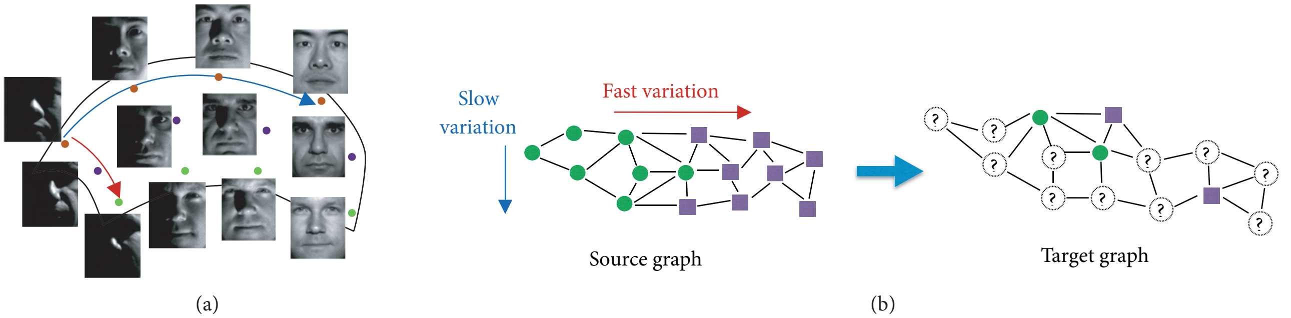


Figure 1. (a) Illustration of a face manifold where the label function varies at different speeds along different directions, (b) Illustration of domain adaptation on graphs

In order to optimize the performance of graph domain adaptation methods, it is important to theoretically characterize their performance limits. The classification performance significantly depends on the structures of the source and the target graphs and the similarity between them. In particular, in problems where the graphs are constructed from data samples coming from data manifolds, the properties of the graphs such as the locations and the weights of the edges, and the number of neighbors of graph nodes largely influence the performance of learning. A thorough characterization of the effects of such parameters in conjunction with the geometry of the data manifolds is necessary to understand the performance limits of graph domain adaptation.

Our contribution in this study is twofold. We first propose a theoretical study of domain adaptation on graphs. We consider a source graph and a target graph constructed with data samples coming respectively from a source manifold and a target manifold. We theoretically analyze the performance of classification on the target graph. In particular, we focus on the estimation error of the target label function and analyze how this error varies with the graph properties, the sampling density of data, and the geometric properties of the data manifolds. Our theoretical analysis suggests that very sparse graphs with too few edges and very dense graphs with too many edges should be avoided, as well as too small edge weights. The smoothness of the label function is shown to positively influence the performance of learning. We show that, under certain assumptions, the estimation error of the target label function decreases with the sampling density of the manifolds at a rate of $O(N^{-1/d})$, where N is the number of samples and d is the intrinsic dimension of the manifolds. Next, we use these theoretical findings to propose a graph domain adaptation algorithm that jointly estimates the class labels of the source and the target data and the topologies of the source and the target graphs. In particular, we optimize the source and the target graph weight matrices, which fully determine the graph topologies, in order to properly control parameters such as the number of neighbors, the minimum edge weights, and the smoothness of the label functions on the graphs. Experimental results on synthetic and real data sets show that the proposed method with learned graph topologies outperforms reference domain adaptation methods with fixed graph topologies and other baseline algorithms.

The rest of the paper is organized as follows. In Section 2, we briefly overview the related literature. In Section 3, we first overview frequency analysis on graphs and then present theoretical bounds for graph domain adaptation. In Section 4, we present a graph domain adaptation algorithm that is motivated by our theoretical findings. We experimentally evaluate the proposed method in Section 5 and conclude in Section 6.

2. Related work

We first overview some common approaches in domain adaptation. In several previous works the covariate shift problem is studied, where the two distributions are matched with reweighting [6, 7]. The studies in [8, 9] propose to train a common classifier after mapping the data to a higher dimensional domain via feature augmentation. Another common approach is to align or match the two domains by mapping them to a common domain with projections or transformations [10–15].

Several domain adaptation methods model data with a graph and make use of the assumption that the label function varies smoothly on the graph [4, 16, 17]. The unsupervised method in [18] formulates the domain adaptation problem as a graph matching problem. The algorithms in [19] and [20] aim to compute a pair of bases on the source graph and the target graph that approximate the Fourier bases and jointly diagonalize the two graph Laplacians. These methods are applied to problems such as clustering and 3D shape analysis. Our recent work [5] also relies on representations with graph Fourier bases, but in the context of domain adaptation. Our study in this paper provides insights for such graph-based methods involving the notion of smoothness on graphs and representations in graph bases.

The problem of learning graph topologies from data has drawn particular interest in recent years. Various efforts have focused on the inference of the graph topology from a set of training signals that are known to vary smoothly on the graph [21–23]. However, such approaches differ from ours in that they address an unsupervised learning problem. Our method, on the other hand, actively incorporates the information of the class labels when learning the graph structures in a domain adaptation framework. In some earlier studies, graph structures were learned in a semisupervised setting [24, 25]. However, unlike our work, the graph Laplacians were restricted to a linear combination of a prescribed set of kernels in these methods.

Some previous studies analyzing the domain adaptation problem from a theoretical perspective are the following. Performance bounds for importance reweighting have been proposed in [6] and [26]. The studies in [27–30] bound the target loss in terms of the deviation between the source distribution and the target distribution. While such studies present a theoretical analysis of domain adaptation, none of them treats the domain adaptation problem in a graph setting. To the best of our knowledge, our theoretical analysis is the first to focus particularly on the graph domain adaptation problem.

3. Theoretical analysis of graph domain adaptation

3.1. Overview of signal processing on graphs

We first briefly overview some basic concepts regarding spectral graph theory and signal processing on graphs [31, 32]. Let $G = (V, E)$ be a graph consisting of N vertices denoted by $V = \{x_i\}_{i=1}^N$ and edges E. The $N \times N$ symmetric matrix W consisting of nonnegative edge weights is called the weight matrix, where W_{ij} is the weight of the edge between the nodes x_i and x_j. If there is no edge between x_i and x_j, then $W_{ij} = 0$. The degree $d_i = \sum_{j=1}^N W_{ij}$ of a vertex x_i is defined as the total weight of the edges linked to it. The diagonal matrix D with entries given by $D_{ii} = d_i$ is called the degree matrix.

A graph signal $f : V \to \mathbb{R}$ is a function that takes a real value on each vertex. A signal f on a graph with N vertices can then be regarded as an N-dimensional vector $f = [f(x_1) \dots f(x_N)]^T \in \mathbb{R}^N$. The graph Laplacian matrix defined as $L = D - W$ is of special importance in spectral graph theory [31, 32]. L can be seen as an operator acting on the function f through the matrix multiplication Lf, and it has been shown to be the graph equivalent of the Laplace operator in Euclidean domains, or the Laplace–Beltrami operator on manifold domains [31, 33, 34]. Recalling that the complex exponentials fundamental in classical signal processing have the special property that they are the eigenfunctions of the Laplace operator, one can extend the notion of frequency to graph domains. Relying on the analogy between the Laplace operator and the graph Laplacian L, one can define a Fourier basis on graphs, which consists of the eigenvectors of L.

Let $u_1, \dots, u_N$ denote the eigenvectors of the graph Laplacian, where $Lu_k = \lambda_k u_k$ for $k = 1, \dots, N$. Here, each u_k is a graph Fourier basis vector of frequency λ_k. The eigenvector u_1 with the smallest eigenvalue $\lambda_1 = 0$ is always a constant function on the graph, and the speed of variation of u_k on the graph increases for increasing k. The eigenvalues $\lambda_1, \dots, \lambda_N$ of the graph Laplacian correspond to frequencies such that λ_k provides a measure of the speed of variation of the signal u_k on the graph. The Fourier basis vectors of an example graph are illustrated in Figure 2. In particular, the speed of variation of a signal f over the graph is

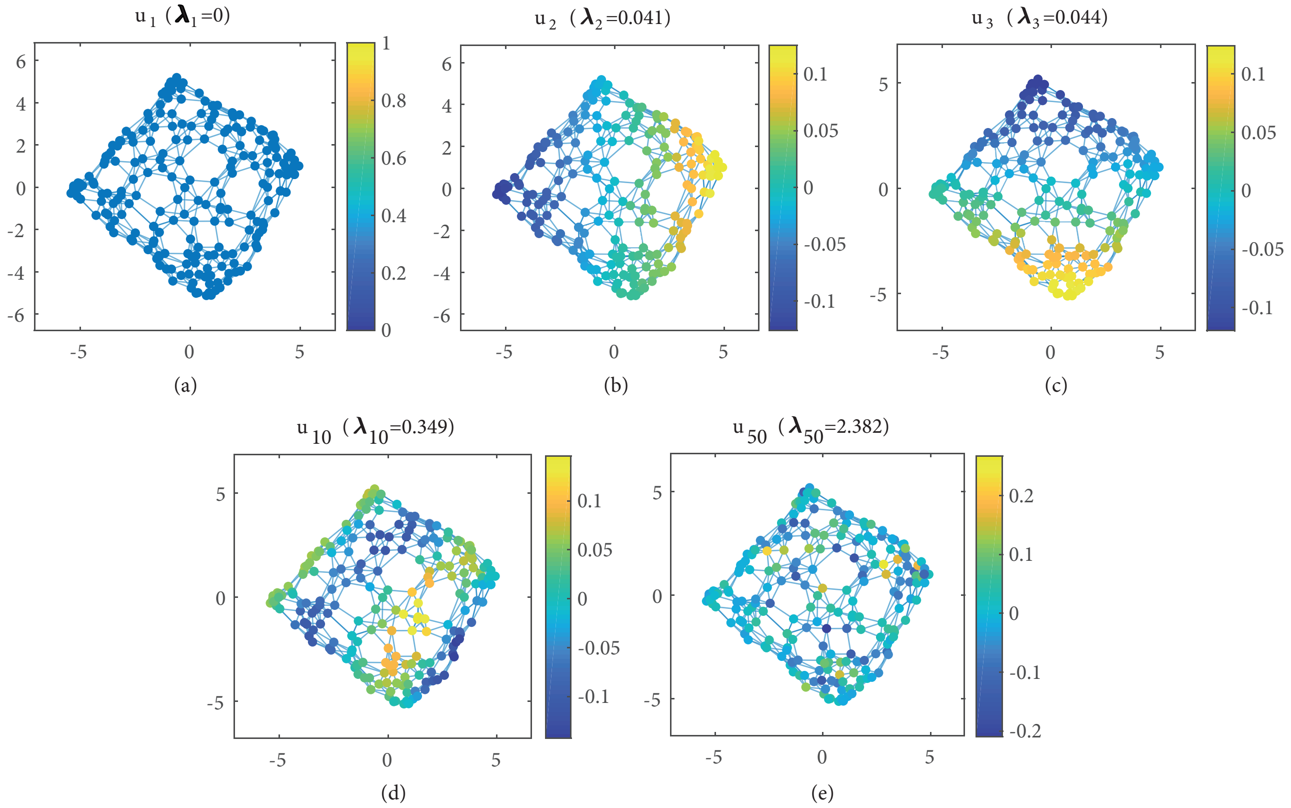
$$f^T L f = \frac{1}{2} \sum_{i,j=1}^{N} W_{ij} (f(x_i) - f(x_j))^2,$$

which takes larger values if the function f varies more abruptly between neighboring graph vertices. Notice that the above term becomes the corresponding eigenvalue λ_k of L when f is taken as u_k, since $u_k^T L u_k = \lambda_k$.

This definition of the graph Fourier basis allows the extension of the Fourier transform to graph domains as follows. Let $U = [u_1 u_2 \dots u_N] \in \mathbb{R}^{N \times N}$ be the matrix consisting of the graph Fourier basis vectors. Then, for a graph signal $f \in \mathbb{R}^N$, the Fourier transform of f can simply be computed as $\alpha = U^T f$, where $\alpha = [\alpha_1 \dots \alpha_N]^T$ is the vector consisting of the Fourier coefficients. Here $\alpha_k = u_k^T f$ is the k-th Fourier coefficient given by the inner product of f and the Fourier basis vector u_k. Note that the graph Fourier basis U is orthonormal as in classical signal processing; hence, the signal f can be reconstructed from its Fourier coefficients as $f = U\alpha$.

3.2. Notation and setting

We now discuss the problem of domain adaptation on graphs and set the notation used in this paper. We consider a source graph $G^s = (V^s, E^s)$ with vertices $V^s = \{x_i^s\}_{i=1}^{N_s}$ and edges E^s; and a target graph $G^t = (V^t, E^t)$ with vertices $V^t = \{x_i^t\}_{i=1}^{N_t}$ and edges E^t. Let W^s and W^t denote the weight matrices, and let L^s and L^t be the Laplacians of the source and the target graphs. Let f^s and f^t be the label functions on the source and



(a)

(b)

(c)

(d)

(e)

Figure 2. Fourier basis vectors of an example graph. The eigenvectors u_1, u_2, u_3, u_{10}, u_{50} of the graph Laplacian are plotted as graph signals in panels (a)–(e), where yellow and blue colors respectively indicate positive and negative values. The first Fourier basis vector u_1 has constant amplitude as its frequency is $\lambda_1 = 0$. The signals u_2 and u_3 have small frequencies around 0.04 and they slowly oscillate along different directions on the graph. The signal u_{10} has a larger frequency $\lambda_{10} = 0.349$; hence, its speed of oscillation is higher. Among the five signals, u_{50} has the highest frequency $\lambda_{50} = 2.382$ and it has the fastest variation on the graph.

the target graphs, which represent class labels in a classification problem and continuously varying entities in a regression problem. We assume that some class labels are known as $y_i^s = f^s(x_i^s)$ for $i \in I_L^s \subset \{1, \dots, N_s\}$, and $y_i^t = f^t(x_i^t)$ for $i \in I_L^t \subset \{1, \dots, N_t\}$, where I_L^s and I_L^t are index sets. Many samples are labeled in the source domain and few samples are labeled in the target domain, i.e. $|I_L^t| \ll |I_L^s|$. Given the available labels $\{y_i^s\}_{i \in I_L^s}$ and $\{y_i^t\}_{i \in I_L^t}$, the purpose of graph domain adaptation is to compute accurate estimates $\hat{f}^s$, $\hat{f}^t$ of f^s and f^t.

All domain adaptation methods rely on a certain relationship between the source and the target domains. In this study, we consider a setting where a relationship can be established between the source and the target graphs through the frequency content of the label functions. Let $f^s = U^s \alpha^s$ and $f^t = U^t \alpha^t$ denote the decompositions of the label functions over the source Fourier basis $U^s = [u_1^s \dots u_{N_s}^s] \in \mathbb{R}^{N_s \times N_s}$ and the target Fourier basis $U^t = [u_1^t \dots u_{N_t}^t] \in \mathbb{R}^{N_t \times N_t}$. We assume a setting where the source and the target label functions have similar spectra and hence similar Fourier coefficients.

We have observed in our previous work [5] that, when computing the estimates $\hat{f}^s$ and $\hat{f}^t$ of the label functions, it is useful to represent them in the reduced bases $\overline{U}^s \in \mathbb{R}^{N_s \times R}$ and $\overline{U}^t \in \mathbb{R}^{N_t \times R}$, which consist of the first R Fourier basis vectors of smallest frequencies. This not only reduces the complexity of the problem but also has a regularization effect since components of very high frequency are excluded from the estimates. The estimates of f^s and f^t are then obtained in the form $\hat{f}^s = \overline{U}^s \overline{\alpha}^s$ and $\hat{f}^t = \overline{U}^t \overline{\alpha}^t$, where $\overline{\alpha}^s \in \mathbb{R}^R$ and $\overline{\alpha}^t \in \mathbb{R}^R$ are reduced Fourier coefficient vectors. In [5], the label functions are estimated such that their Fourier coefficients $\overline{\alpha}^s$ and $\overline{\alpha}^t$ are close to each other and the estimates $\hat{f}^s$ and $\hat{f}^t$ are consistent with the available labels.

In our theoretical analysis of graph domain adaptation, we consider a setting where graph nodes are sampled from data manifolds. Let $\{x_i^s\}_{i=1}^{N_s} \subset \mathcal{M}^s$ and $\{x_i^t\}_{i=1}^{N_t} \subset \mathcal{M}^t$ denote source and target graph nodes sampled from a source data manifold $\mathcal{M}^s$ and a target data manifold $\mathcal{M}^t$. We assume that the source and the target data manifolds are generated through a pair of functions $g^s : \Gamma \rightarrow \mathcal{M}^s$ and $g^t : \Gamma \rightarrow \mathcal{M}^t$ defined on a common parameter space Γ. Then, each manifold sample can be expressed as $x_i^s = g^s(\gamma_i^s)$ and $x_i^t = g^t(\gamma_i^t)$, where $\gamma_i^s \in \Gamma$ and $\gamma_i^t \in \Gamma$ are parameter vectors as illustrated in Figure 3. The parameter vectors are assumed to capture the source of variation generating the data manifolds. For instance, in a face recognition problem, γ_i^s and γ_i^t may represent rotation angles of the cameras viewing the subjects; and the discrepancy between $\mathcal{M}^s$ and $\mathcal{M}^t$ may result from the change in the illumination conditions. Note that in domain adaptation no relation is assumed to be known between γ_i^s and γ_i^t. Moreover, the parameters $\{\gamma_i^s\}$, $\{\gamma_i^t\}$ and the functions g^s, g^t are often not known in practice. Although we consider this setting in our theoretical analysis, the practical algorithm we propose in Section 4 will not require the knowledge of these parameters.

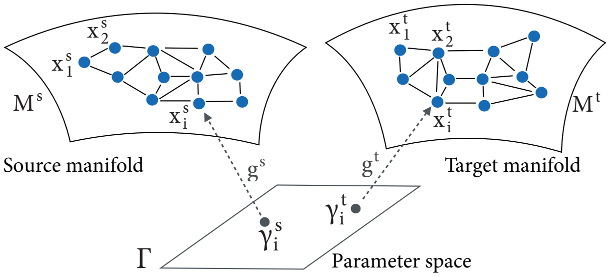


Figure 3. Illustration of the domain adaptation setting considered in our study

3.3. Performance bounds for graph domain adaptation

In this section, we analyze the error between the estimated target label function $\hat{f}^t$ and the true target label function f^t. We would like to derive an upper bound for the estimation error

$$E = \|\hat{f}^t - f^t\|^2 = \sum_{i=1}^{N_t} (\hat{f}^t(x_i^t) - f^t(x_i^t))^2 = \sum_{i=1}^{N_t} (\hat{f}_i^t - f_i^t)^2$$

where $\hat{f}_i^t = \hat{f}^t(x_i^t)$ and $f_i^t = f^t(x_i^t)$ simply denote the values that the estimated and the true label functions take at the sample x_i^t.

We first define some parameters regarding the properties of the data manifolds. For the convenience of analysis, we assume that the source and the target graphs contain equally many samples[1], i.e. $N_s = N_t = N$.

[1] This assumption is made for simplifying the theoretical analysis. The algorithm proposed in Section 4 does not require the source and the target graphs to have an equal number of nodes.

We assume that the manifolds $\mathcal{M}^s \subset \mathcal{H}^s$ and $\mathcal{M}^t \subset \mathcal{H}^t$ are embedded in the Hilbert spaces $\mathcal{H}^s$, $\mathcal{H}^t$; the parameter space Γ is a Banach space, and the manifolds $\mathcal{M}^s$ and $\mathcal{M}^t$ have (intrinsic) dimension d.

We assume that the functions $g^s : \Gamma \rightarrow \mathcal{M}^s$ and $g^t : \Gamma \rightarrow \mathcal{M}^t$ are Lipschitz-continuous, respectively with constants M_s and M_t; i.e. for any two parameter vectors $\gamma_1, \gamma_2 \in \Gamma$, we assume that

$$\|g^s(\gamma_1) - g^s(\gamma_2)\| \leq M_s \|\gamma_1 - \gamma_2\|, \qquad \|g^t(\gamma_1) - g^t(\gamma_2)\| \leq M_t \|\gamma_1 - \gamma_2\|$$

where $\|\cdot\|$ denotes the usual norm in the space of interest. Thus, the constants M_s and M_t provide a measure of smoothness for the manifolds $\mathcal{M}^s$ and $\mathcal{M}^t$. We further assume that there exist two constants A_l and A_u such that for any $\gamma_1 \neq \gamma_2$ in Γ,

$$A_l \leq \frac{\|g^t(\gamma_1) - g^t(\gamma_2)\|}{\|g^s(\gamma_1) - g^s(\gamma_2)\|} \leq A_u. \tag{1}$$

The constants A_l and A_u indicate the similarity between the geometric structures of the manifolds $\mathcal{M}^s$ and $\mathcal{M}^t$. As the variations of the functions g^s and g^t over Γ become more similar, the constants A_l and A_u get closer to 1. Let $A = \max(|1 - A_l|, |A_u - 1|)$ denote a bound on the deviations of A_l and A_u from 1.

We consider a setting where the graph weight matrices are obtained with a kernel ϕ such that $W^s_{ij} = \phi(\|x^s_i - x^s_j\|)$ and $W^t_{ij} = \phi(\|x^t_i - x^t_j\|)$ for neighboring samples on the graph. We assume that the kernel $\phi : \mathbb{R}^+ \cup \{0\} \rightarrow \mathbb{R}^+$ is an L_ϕ-Lipschitz nonincreasing function with $|\phi(u) - \phi(v)| \leq L_\phi |u - v|$ for any $u, v \in \mathbb{R}^+ \cup \{0\}$. Let us denote the maximum value of the kernel function as $\phi_0 := \phi(0)$.

In our analysis, we consider domain adaptation algorithms that compute an estimate $\hat{f}^t$ of f^t whose values at the labeled nodes agree with the given labels, i.e. $\hat{f}^t_i = f^t_i$ for $i \in I^t_L$. Let $w^{\min}$, $K^{\max}$, and $K^{\min}$ be parameters representing the smallest edge weight, the maximum number of neighbors, and the minimum number of neighbors in the target graph. A more precise description of these parameters can be found in the accompanying technical report [35, Appendix A].

We are now ready to state our main result in the following theorem.

Theorem 1 *Consider a graph domain adaptation algorithm that estimates the source and the target label functions as* $\hat{f}^s = \overline{U}^s \overline{\alpha}^s$ *and* $\hat{f}^t = \overline{U}^t \overline{\alpha}^t$ *such that the difference between their Fourier coefficients is bounded as* $\|\overline{\alpha}^s - \overline{\alpha}^t\| \leq \Delta_\alpha$, *the norms of the Fourier coefficients are bounded as* $\|\overline{\alpha}^s\|, \|\overline{\alpha}^t\| \leq C$, *and* $\hat{f}^s$ *and* $\hat{f}^t$ *are band-limited on the graphs so as not to contain any components with frequencies larger than* λ_R. *Assume that the estimate* $\hat{f}^s$ *is equal to the true source label function* f^s *(e.g., as in a setting where all source samples are labeled). Then, the target label estimation error can be upper bounded as*

$$\|\hat{f}^t - f^t\|^2 \leq \frac{\kappa}{w^{\min}} (\sqrt{B} + \sqrt{\hat{B}})^2 \tag{2}$$

where B *is an upper bound on the rate of variation of the true label function*

$$(f^t)^T L^t f^t \leq B, \tag{3}$$

the parameter κ *is a function of the minimum and maximum number of neighbors of the form*

$$\kappa = O\left(\frac{1}{K^{\min}} \operatorname{poly}\left(\frac{K^{\max}}{K^{\min}} \right) \right), \tag{4}$$

with $\text{poly}(\cdot)$ *denoting polynomial dependence,* $\hat{B}$ *is an upper bound on the speed of variation of the target label estimate given by*

$$(\hat{f}^t)^T L^t \hat{f}^t \leq \hat{B} := (f^s)^T L^s f^s + C^2 \rho_{\max} + 2C\lambda_R \Delta_\alpha, \tag{5}$$

$\rho_{\max}$ *is a geometry-dependent parameter varying at rate*

$$\rho_{\max} := O(L_\phi \, (A \, M_s + M_s + M_t)\epsilon_\Gamma + \phi_0), \tag{6}$$

and ϵ_Γ *is proportional to the largest parameter-domain distance between neighboring graph nodes.*

Theorem 1 is stated more precisely in [35, Appendix A], where its proof can also be found. In the proof, first an upper bound is derived on the difference between the rates of variation of the source and the target label functions. Next, the deviation between the eigenvalues of the source and the target graph Laplacians is studied. Finally, these two results are combined to obtain an upper bound on the estimation error of the target label function.

Theorem 1 can be interpreted as follows. First, observe that the estimation error increases linearly with the bound Δ_α on the deviation between the source and the target Fourier coefficients. This suggests that in graph domain adaptation it is favorable to estimate the source and the target label functions in a way that they have similar spectra. The theorem also has several implications regarding the smoothness of label functions and their estimates. It is well known that graph-based learning methods perform better if label functions vary smoothly on the graph. This is formalized in Theorem 1 via the assumption that the estimates $\hat{f}^s$ and $\hat{f}^t$ are band-limited such that the highest frequency present in their spectrum (computed with the graph Fourier transform) does not exceed some threshold λ_R, which limits their speeds of variation on the graphs. Notice that the rates of variation $(f^s)^T L^s f^s$ and $(f^t)^T L^t f^t$ of the true source and target label functions also affect the estimation error through the terms B and $\hat{B}$.

Next, we observe from Eq. (2) that the estimation error depends on the geometric properties of data manifolds as follows. The error increases linearly with $\rho_{\max}$ (through the term $\hat{B}$), while in Eq. (6), $\rho_{\max}$ is seen to depend linearly on the parameters A, M_s, and M_t. Recalling that M_s and M_t are the Lipschitz constants of the functions g^s and g^t defining the data manifolds, the theorem suggests that the estimation error is smaller when the data manifolds are smoother. The fact that the error increases linearly with the parameter A is intuitive as A captures the dissimilarity between the geometric structures of the source and the target manifolds. We also notice that $\rho_{\max}$ is proportional to the parameter ϵ_Γ. We give a precise definition of the parameter ϵ_Γ in [35], which can be roughly described as an upper bound on the parameter-domain (Γ) distance between neighboring graph samples. As the number of samples N increases, ϵ_Γ decreases at rate $\epsilon_\Gamma = O(N^{-1/d})$, where d is the intrinsic dimension of the manifolds. Since $\rho_{\max}$ is linearly proportional to ϵ_Γ, we conclude that the estimation error of the target label function decreases with N at the same rate $O(N^{-1/d})$. This can be intuitively interpreted in the way that the discrepancy between the topologies of the source and the target graphs resulting from finite sampling effects decreases as the sampling of the data manifolds becomes denser.

The result in Theorem 1 also leads to the following important conclusions about the effect of the graph properties on the performance of learning. First, the estimation error is observed to increase linearly with the parameter-domain distance ϵ_Γ between neighboring points on the graphs. This suggests that when constructing the graphs, two samples that are too distant from each other in the parameter space should rather not be connected with an edge. Then, we notice that the parameter κ decreases when the ratio $K^{\max}/K^{\min}$ between the maximum and the minimum number of neighbors is smaller. Hence, more “balanced” graph topologies influence the performance positively; more precisely, the number of neighbors of different graph nodes should not be disproportionate. At the same time, the term $K^{\min}$ in the denominator in Eq. (4) implies that nodes with too few neighbors should rather be avoided. A similar observation can be made about the term $w^{\min}$ in the expression of the error bound in Eq. (2). The minimal edge weight term $w^{\min}$ in the denominator suggests that graph edges with too small weights have the tendency to increase the error. From all these observations, we conclude that when constructing graphs, balanced graph topologies should be preferred and significant variation of the number of neighbors across different nodes, too isolated nodes, and too weak edges should be avoided.

4. Learning graph topologies for domain adaptation

In this section, we propose an algorithm for jointly learning graphs and label functions for domain adaptation, based on the theoretical findings presented in Section 3. We first formulate the graph domain adaptation problem and then propose a method for solving it.

4.1. Problem formulation

Given the source and the target samples $\{x_i^s\}_{i=1}^{N_s} \subset \mathbb{R}^n$ and $\{x_i^t\}_{i=1}^{N_t} \subset \mathbb{R}^n$, we consider the problem of constructing a source graph and a target graph with respective vertex sets $\{x_i^s\}$ and $\{x_i^t\}$, while obtaining estimates $\hat{f}^s = \overline{U}^s \overline{\alpha}^s$ and $\hat{f}^t = \overline{U}^t \overline{\alpha}^t$ of the label functions at the same time. The problem of learning the graph topologies is equivalent to the problem of learning the weight matrices W^s and W^t.

The bound in Eq. (2) on the target error suggests that when learning a pair of graphs, the parameters κ, B, and $\hat{B}$ should be kept small, whereas small values for $w^{\min}$ should be avoided. The expression in Eq. (5) shows that the parameters λ_R and Δ_α should be kept small. Meanwhile, in the expression of $\rho_{\max}$ in Eq. (6), we observe that the terms A, M_s, and M_t are determined by the geometry of the data manifolds and these are independent of the graphs. On the other hand, the parameter ϵ_Γ depends on the graph topology and can be controlled more easily. Thus, in view of the interpretation of Theorem 1, we propose to jointly learn the label functions $\hat{f}^s$ and $\hat{f}^t$ and the weight matrices W^s and W^t based on the following optimization problem.

$$\begin{aligned}
&\text{minimize}_{\overline{\alpha}^s, \overline{\alpha}^t, W^s, W^t} \ \|S^s \overline{U}^s \overline{\alpha}^s - y^s\|^2 + \|S^t \overline{U}^t \overline{\alpha}^t - y^t\|^2 + \mu \|\overline{\alpha}^s - \overline{\alpha}^t\|^2 \\
&\qquad + (\hat{f}^s)^T L^s \hat{f}^s + (\hat{f}^t)^T L^t \hat{f}^t + \mu_s \sum_{i,j=1}^{N_s} W_{ij}^s \|x_i^s - x_j^s\|^2 + \mu_t \sum_{i,j=1}^{N_t} W_{ij}^t \|x_i^t - x_j^t\|^2 \\
&\text{subject to } W_{ij}^s \geq W^{\min}, \ \forall i,j \in \{1,\dots,N_s\} \text{ with } W_{ij}^s \neq 0; \quad W_{ij}^t \geq W^{\min}, \ \forall i,j \in \{1,\dots,N_t\} \text{ with } W_{ij}^t \neq 0; \\
&\qquad d_{\min} \leq d_i^s \leq d_{\max}, \ \forall i \in \{1,\dots,N_s\}; \quad d_{\min} \leq d_i^t \leq d_{\max}, \ \forall i \in \{1,\dots,N_t\}.
\end{aligned} \tag{7}$$

Here μ, μ_s, and μ_t are positive weight parameters, and y^s and y^t are vectors consisting of all the available source and target labels. The first two terms $\|S^s \overline{U}^s \overline{\alpha}^s - y^s\|^2$ and $\|S^t \overline{U}^t \overline{\alpha}^t - y^t\|^2$ in (7) enforce the estimated label functions $\hat{f}^s$ and $\hat{f}^t$ to be consistent with the available labels, where S^s and S^t are binary selection matrices consisting of 0’s and 1’s that select the indices of labeled data. The third term $\|\overline{\alpha}^s - \overline{\alpha}^t\|^2$ aims to reduce the parameter Δ_α in (5). Note that the representation of $\hat{f}^s$ and $\hat{f}^t$ in terms of the first R Fourier basis vectors in $\overline{U}^s$ and $\overline{U}^t$ is useful for keeping the parameter λ_R small.

Next, the minimization of the terms $(\hat{f}^s)^T L^s \hat{f}^s$ and $(\hat{f}^t)^T L^t \hat{f}^t$ encourages the label functions $\hat{f}^s$ and $\hat{f}^t$ to vary slowly on the graphs, which aims to reduce the parameters $\hat{B}$ and B in Eqs. (5) and (3). We also recall from Theorem 1 that in order to make the parameter ϵ_Γ small, graph edges between distant points should be avoided. The terms $\sum_{i,j} W^s_{ij} \|x^s_i - x^s_j\|^2$ and $\sum_{i,j} W^t_{ij} \|x^t_i - x^t_j\|^2$ aim to achieve this by penalizing large edge weights between distant samples. The inequality constraints $W^s_{ij} \geq W^{\min}$ and $W^t_{ij} \geq W^{\min}$ on nonzero edge weights ensure that the minimum edge weight $w^{\min}$ is above some predetermined threshold $W^{\min}$.

Finally, we recall from Theorem 1 that in order to minimize κ, the ratio $K^{\max}/K^{\min}$ must be kept small while avoiding too small $K^{\min}$ values. However, incorporating the number of neighbors directly in the objective function would lead to an intractable optimization problem. Noticing that the node degrees are expected to be proportional to the number of neighbors, we prefer to relax this to the constraints $d_{\min} \leq d^s_i \leq d_{\max}$ and $d_{\min} \leq d^t_i \leq d_{\max}$ on the node degrees, where d^s_i and d^t_i respectively denote the degrees of x^s_i and x^t_i; and $d_{\min}$ and $d_{\max}$ are some predefined degree threshold parameters with $0 < d_{\min} \leq d_{\max}$.

4.2. Proposed Method: domain adaptive graph learning

Analyzing the optimization problem in Eq. (7), we observe that the matrices $\overline{U}^s$ and $\overline{U}^t$ are nonconvex and highly nonlinear functions of the optimization variables W^s and W^t as they consist of the eigenvectors of the graph Laplacians L^s and L^t. Moreover, due to the multiplicative terms such as $\overline{U}^s \overline{\alpha}^s$, the problem is even not jointly convex in $\overline{\alpha}^s$, $\overline{\alpha}^t$, $\overline{U}^s$, and $\overline{U}^t$. Hence, it is quite difficult to solve the problem in Eq. (7). In our method, we employ a heuristic iterative solution approach that relaxes the original problem in Eq. (7) into more tractable subproblems and alternatively updates the coefficients and the weight matrices in each iteration as follows.

We first initialize the weight matrices W^s, W^t with a typical strategy; e.g., by connecting each node to its K nearest neighbors and assigning edge weights with a Gaussian kernel. We use the normalized versions of the graph Laplacians given by $L^s = (D^s)^{-1/2}(D^s - W^s)(D^s)^{-1/2}$ and $L^t = (D^t)^{-1/2}(D^t - W^t)(D^t)^{-1/2}$.

In the first step of each iteration, we optimize $\overline{\alpha}^s$, $\overline{\alpha}^t$ by fixing the weight matrices W^s, W^t. This gives the following optimization problem:

$$\text{minimize}_{\overline{\alpha}^s, \overline{\alpha}^t} \|S^s \overline{U}^s \overline{\alpha}^s - y^s\|^2 + \|S^t \overline{U}^t \overline{\alpha}^t - y^t\|^2 + \mu \|\overline{\alpha}^s - \overline{\alpha}^t\|^2. \tag{8}$$

The simplified objective[2] in Eq. (8) is in fact the same as the objective of the SDA algorithm proposed in [5]. As the problem is quadratic and convex in $\overline{\alpha}^s$ and $\overline{\alpha}^t$, its solution can be analytically found by setting the gradient equal to 0, which gives [5]:

$$\overline{\alpha}^s = (\mu^{-1} A^t A^s + A^t + A^s)^{-1}(\mu^{-1} A^t B^s y^s + B^s y^s + B^t y^t), \qquad \overline{\alpha}^t = (\mu^{-1} A^s \overline{\alpha}^s + \overline{\alpha}^s - \mu^{-1} B^s y^s)$$

where $A^s = (\overline{U}^s)^T (S^s)^T S^s \overline{U}^s$, $\quad B^s = (\overline{U}^s)^T (S^s)^T$, $\quad A^t = (\overline{U}^t)^T (S^t)^T S^t \overline{U}^t$, $\quad B^t = (\overline{U}^t)^T (S^t)^T$.

[2] Note that the dependence of $\hat{f}^s$ and $\hat{f}^t$ on $\overline{\alpha}^s$ and $\overline{\alpha}^t$ is neglected in Eq. (8). The reason is that since $\overline{U}^s$ and $\overline{U}^t$ consist of the eigenvectors of L^s and L^t, the terms $(\hat{f}^s)^T L^s \hat{f}^s$ and $(\hat{f}^t)^T L^t \hat{f}^t$ would contribute to the objective only as regularization terms on the weighted norms of $\overline{\alpha}^s$ and $\overline{\alpha}^t$. We prefer to exclude such a regularization in order to prioritize fitting the coefficients $\overline{\alpha}^s$, $\overline{\alpha}^t$ to each other and to the available labels.

Algorithm 1 Spectral domain adaptation via domain adaptive graph learning (SDA-DAGL)

```
Input:
  $\{x_i^s\}$, $\{x_i^t\}$: Source and target samples
  $y^s$, $y^t$: Available source and target labels
Initialization:
  Initialize the weight matrices $W^s$, $W^t$ with a sufficiently large number of edges, e.g., as K-NN graphs.
repeat
    Compute the graph Laplacians $L^s$, $L^t$ and Fourier bases $\overline{U}^s$, $\overline{U}^t$ with weight matrices $W^s$, $W^t$.
    Update coefficients $\overline{\alpha}^s$, $\overline{\alpha}^t$ by solving Eq. (8).
    Update the weight matrices $W^s$, $W^t$ by solving Eq. (9) via linear programming.
    Prune the graph edges by setting the edge weights with $W_{ij}^s < W^{\min}$ and $W_{ij}^t < W^{\min}$ to 0.
until the maximum number of iterations is attained
Output:
  $f^t = \overline{U}^t \overline{\alpha}^t$: Estimated target label function
  $f^s = \overline{U}^s \overline{\alpha}^s$: Estimated source label function
```

Then, in the second step of an iteration, we fix the coefficients $\overline{\alpha}^s$ and $\overline{\alpha}^t$, and optimize the weight matrices W^s and W^t. As the dependence of the Fourier basis matrices $\overline{U}^s$ and $\overline{U}^t$ on W^s and W^t is quite intricate, we fix $\overline{U}^s$ and $\overline{U}^t$ to their values from the preceding iteration and neglect this dependence when reformulating our objective for learning the weight matrices. Defining the vectors $\hat{h}^s = (D^s)^{-1/2}\hat{f}^s$ and $\hat{h}^t = (D^t)^{-1/2}\hat{f}^t$, the fourth and fifth terms in Eq. (7) can be rewritten as

$$(\hat{f}^s)^T L^s \hat{f}^s + (\hat{f}^t)^T L^t \hat{f}^t = (\hat{h}^s)^T (D^s - W^s)\hat{h}^s + (\hat{h}^t)^T (D^t - W^t)\hat{h}^t.$$

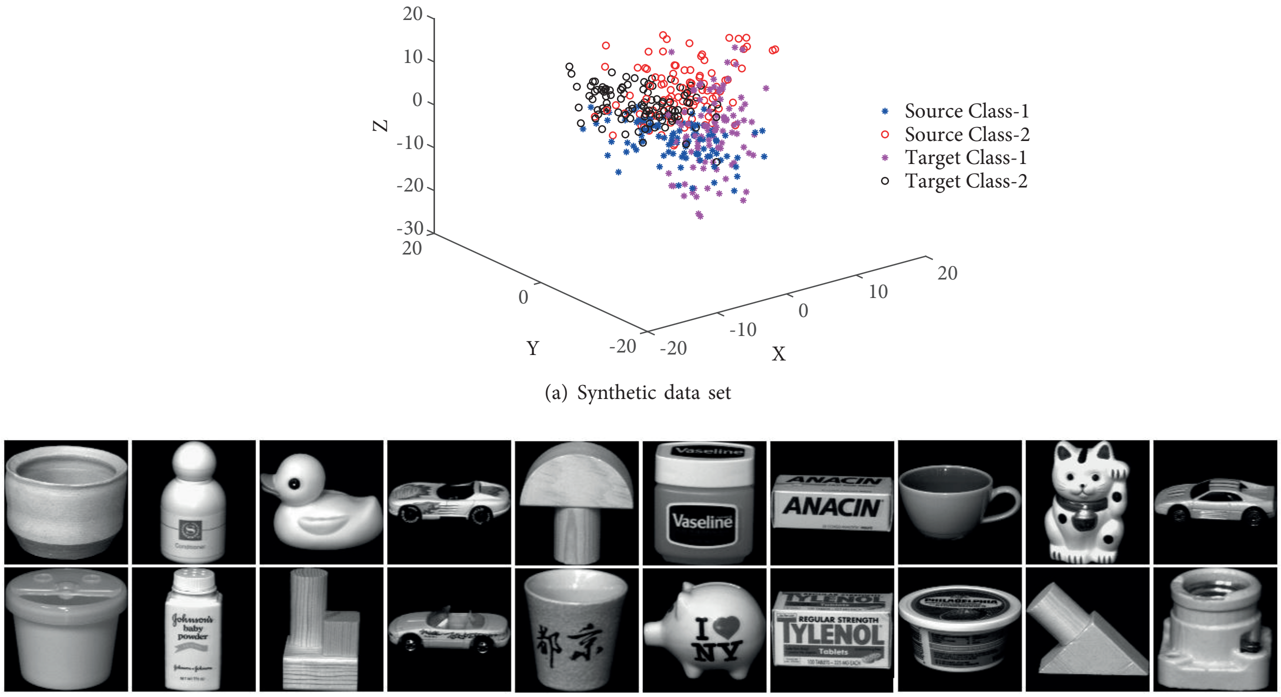
If the node degrees were fixed, the minimization of the above term would correspond to the maximization of $(\hat{h}^s)^T W^s \hat{h}^s + (\hat{h}^t)^T W^t \hat{h}^t$. However, we have observed that letting the node degrees vary in an appropriate interval gives better results than fixing them. We thus propose the following problem for optimizing W^s and W^t:

$$\begin{aligned}
&\text{minimize}_{\,W^s,W^t} \quad \mu_s \sum_{i,j=1}^{N_s} W_{ij}^s \|x_i^s - x_j^s\|^2 - (\hat{h}^s)^T W^s \hat{h}^s + \mu_t \sum_{i,j=1}^{N_t} W_{ij}^t \|x_i^t - x_j^t\|^2 - (\hat{h}^t)^T W^t \hat{h}^t \\
&\text{subject to } d_{\text{min}} \le \sum_{j=1}^{N_s} W_{ij}^s \le d_{\text{max}}, \ \forall i = 1, \dots, N_s; \quad 0 \le W_{ij}^s \le 1, \ \forall i,j = 1, \dots, N_s; \\
&\qquad d_{\text{min}} \le \sum_{j=1}^{N_t} W_{ij}^t \le d_{\text{max}}, \ \forall i = 1, \dots, N_t; \quad 0 \le W_{ij}^t \le 1, \ \forall i,j = 1, \dots, N_t.
\end{aligned} \tag{9}$$

The objective function and the constraints of the problem in Eq. (9) are linear in the entries of W^s and W^t. Hence, Eq. (9) can be posed as a linear programming (LP) problem and can be solved with an LP solver. In the problem in Eq. (9), the edge weights are constrained to lie between 0 and 1. Since the solution of an LP problem occurs at a corner point of the feasible region, many entries of the weight matrices solving the LP problem in Eq. (9) are 0. This gradually improves the sparsity of the weight matrices. Then, instead of directly incorporating the sparsity of the weight matrices in the optimization problem and imposing a lower bound on the positive edge weights as in the original problem in Eq. (7), we prefer to initialize the graphs with a sufficiently high number of edges, solve the LP problem in Eq. (9) by optimizing only the nonzero edge weights, and then at the end of each iteration apply a graph pruning step that sets the edge weights smaller than $W^{\min}$ to 0. After each iteration, the graph Laplacians L^s, L^t and the Fourier bases $\overline{U}^s$, $\overline{U}^t$ are updated. The same procedure then continues with the optimization of the Fourier coefficients as in Eq. (8). We call this algorithm spectral domain adaptation via domain adaptive graph learning (SDA-DAGL) and summarize it in Algorithm 1. Due to the various relaxations made in different stages, it is not possible to guarantee the convergence of the solution in general. In practice, we have found it useful to terminate the algorithm after a suitably chosen number of iterations.

5. Experimental results

We now evaluate the proposed method with experiments on a synthetic data set and a real data set. The synthetic data set shown in Figure 4a consists of 400 normally distributed samples in $\mathbb{R}^3$ from two classes. The two classes in each domain have different means, and the source and the target domains differ by a rotation of 90°. The variance of the normal distributions is chosen to be relatively large to ensure a sufficient level of difficulty. The COIL-20 object database [37] shown in Figure 4b consists of a total of 1440 images of 20 objects. Each object has 72 images taken from different camera angles rotating around it. We downsample the images to a resolution of 32×32 pixels. The 20 objects in the data set are divided into two groups and each object in the first group is matched to the object in the second group that is the most similar to it. Each group of 10 objects is taken as a different domain and the matched object pairs are considered to have the same class label in the experiments.



(a) Synthetic data set

(b) COIL-20 data set

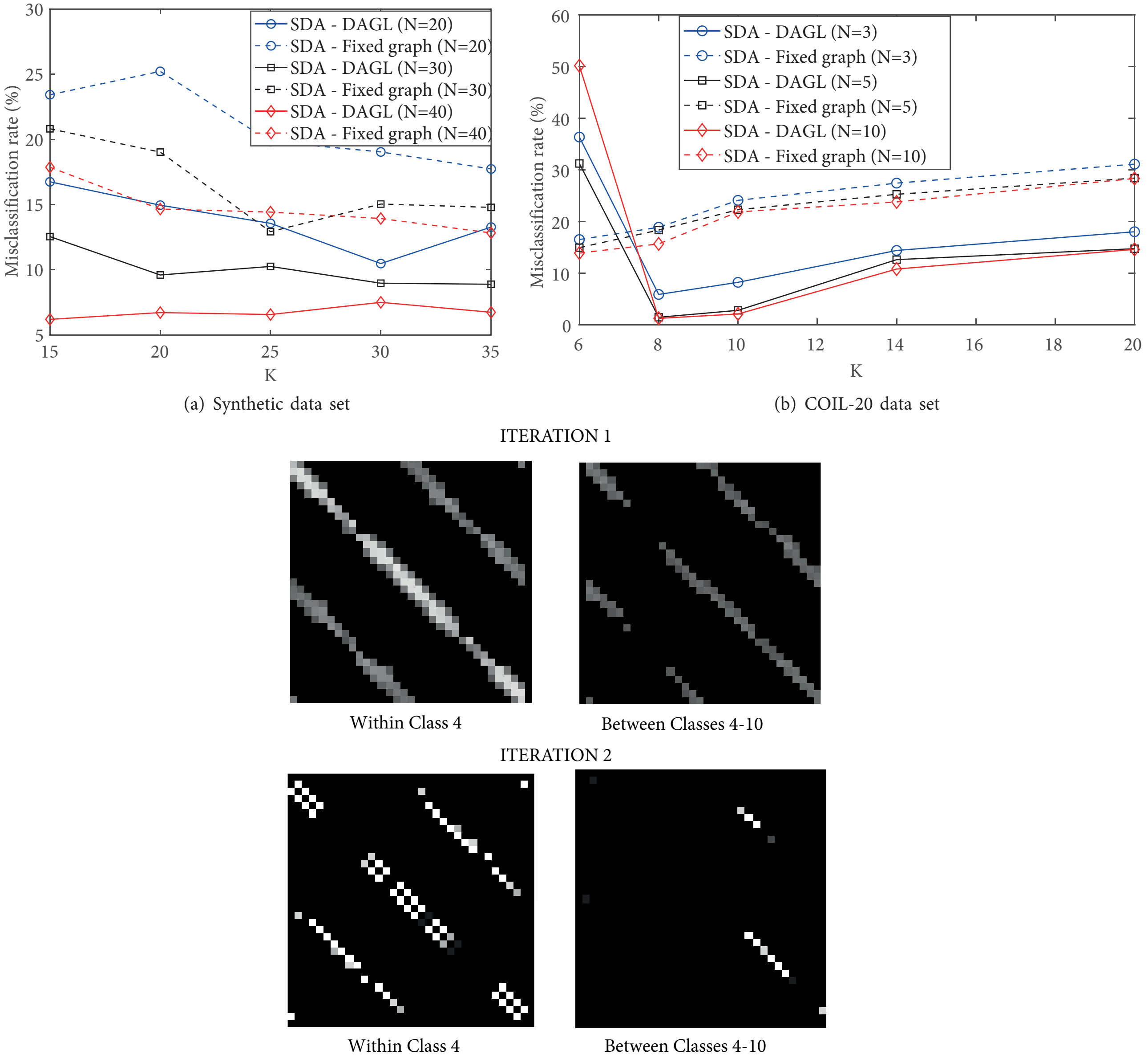
Figure 4. (a) Synthetic data set with two classes drawn from normal distributions. (b) Sample images from the COIL-20 data set. Each source object in the upper row is assigned the same class label as the target object right below it.

We first compare our SDA-DAGL algorithm with the SDA algorithm in order to study the efficiency of the proposed graph learning approach. In both data sets, we first independently construct the source and the target graphs by connecting each sample to its K nearest neighbors and forming the weight matrices W^s and W^t with a Gaussian kernel. The SDA algorithm uses the fixed graph topology represented by these weight matrices. In the proposed SDA-DAGL method, these weight matrices are used to initialize the algorithm as in Step 2 of Algorithm 1, and then they are refined gradually with the proposed joint graph learning and label estimation approach. All class labels are known in the source domain, while a small number of labels are known in the target domain. The class labels of the unlabeled target samples are estimated with the two algorithms in comparison. Figure 5 shows the variation of the misclassification rate of target samples with the number of neighbors (K) for different numbers of labeled samples (N) in the target domain. The results are averaged over around 10 random repetitions of the experiment with different selections of the labeled samples.

The results in Figure 5 show that the SDA-DAGL algorithm performs better than the SDA algorithm in almost all cases. This suggests that even if the SDA-DAGL algorithm is initialized with nonoptimal graphs, it can successfully learn a suitable pair of source and target graphs and accurately estimate the target labels. The performance of SDA-DAGL is seen to be robust to the choice of the initial number of neighbors K in the synthetic data set in Figure 5a, whereas it is more affected by the choice of K in the COIL-20 data set in Figure 5b. This is because the COIL-20 data set conforms quite well to a low-dimensional manifold structure due to the regular sampling of the camera rotation parameter generating the data set. Initializing the weight matrices with too high K values leads to the loss of the information of the geometric structure from the beginning and makes it more difficult for the algorithm to recover the correct graph topologies along with the label estimates. The fact that too small K values also yield large error in Figure 5b can be explained with the incompatibility of small K values with the graph pruning strategy employed in our method. We also show, in Figure 5c, the evolution of the weight matrix W^t during two consecutive iterations of the SDA-DAGL method for the COIL-20 data set. The weights of the within-class edges in Class 4 and the between-class edges between Classes 4 and 10 are shown. The update on W^t is seen to mostly preserve the within-class edges, while it removes most of the between-class edges. Thus, the learned graph topology is progressively improved.

We then study how the difference between the sizes of the source and the target graphs affects the algorithm performance. We fix the number N_s of source nodes, and vary the number N_t of target nodes by constructing the target graph with a randomly selected subset of the target samples. The variation of the classification error with N_t for different N values (number of labeled target nodes) is presented in Table. The results show that the best performance is obtained when the source and the target graphs have an equal number of nodes ($N_t = N_s$). In the synthetic data set, the removal of up to 25% of the target graph nodes is seen to be tolerable without much loss in the performance. On the other hand, the performance degrades more severely in the COIL-20 data set as the difference between the source and the target graph sizes increases. Due to the very particular geometric structure of this data set, the algorithm is more sensitive to the dissimilarity between the graph topologies.

We finally present an overall comparison of the proposed SDA-DAGL method with some baseline domain adaptation methods representing different approaches. We compare our method to the Easy Adapt ++ [8] algorithm based on feature augmentation; the domain adaptation using manifold alignment [36] algorithm which is a graph-based method learning a supervised embedding; and the geodesic flow Kernel [12], and subspace alignment [11] methods, which align the PCA bases of the two domains via unsupervised projections. The misclassification rates of the algorithms on target samples are plotted with respect to the ratio of known



(a) Synthetic data set

(b) COIL-20 data set

(c) Target weight matrix

Figure 5. (a), (b) The variation of the misclassification rates of the target samples with the number of nearest neighbors K. The curves with dashed lines are obtained with a fixed graph topology (SDA), whereas the corresponding curves with solid lines are obtained with the proposed graph learning method (SDA-DAGL). The results show that the graphs learnt with the proposed method yield higher performance than fixed graph topologies constructed with K-NN. (c) The evolution of W^t during two consecutive iterations. The black color represents 0 weight (no edge) and brighter tones indicate larger edge weights. The updates in the graph topology tend to preserve within-class edges and suppress between-class edges as desired.

target labels in Figures 6a and 6b, respectively for the synthetic data set and the COIL-20 data sets. The misclassification error decreases as the ratio of the known target labels increases as expected. In both data sets, the proposed SDA-DAGL algorithm is often observed to outperform the baseline approaches and the SDA method, which uses a fixed graph topology. This suggests that the proposed graph learning strategy provides an effective solution for improving the performance of domain adaptation on graphs.

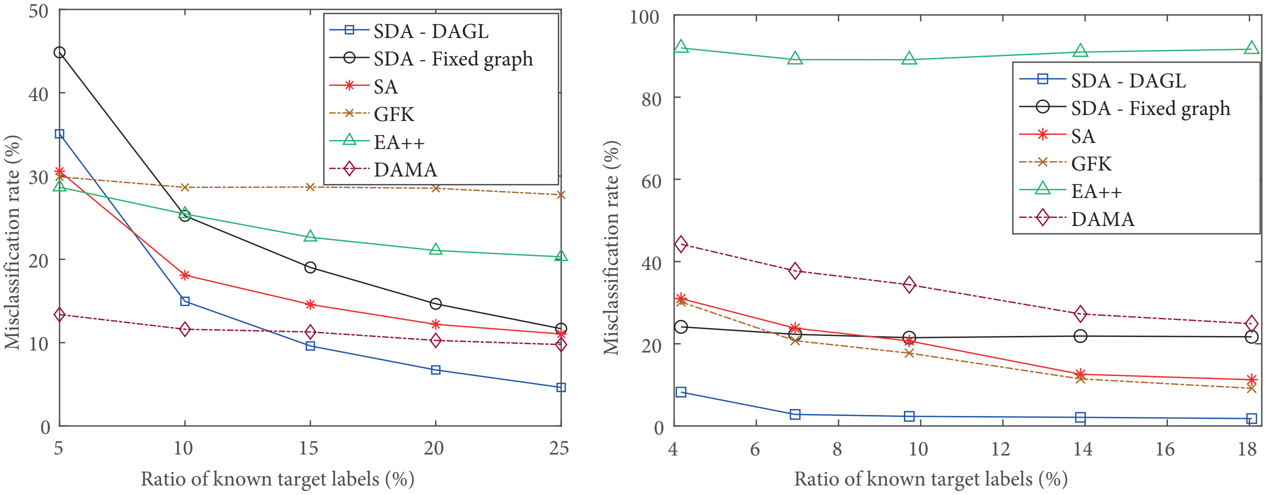
Table. The variation of the target classification error with the number of target graph nodes. The classification performance tends to be higher when the source and the target graph sizes are similar.

N_t	100	125	150	175	200
$N = 20$	17.13	12.67	5.31	8.51	6.67
$N = 30$	11.57	8.52	5.50	6.62	6.17
$N = 40$	10.17	5.53	4.82	6.81	4.31

(a) Synthetic data set, $N_s = 200$

N_t	30	40	50	60	72
$N = 3$	28.22	25.08	22.42	17.79	6.52
$N = 5$	27.44	24.11	15.60	14.15	1.13
$N = 10$	22.70	18.67	16.85	5.76	1.06

(b) COIL-20 data set, $N_s = 72$



(a) Synthetic data set

(b) COIL-20 data set

Figure 6. The variation of the misclassification rate of the target samples with the ratio of labeled target samples. The proposed method is seen to outperform baseline approaches in most cases.

6. Conclusion

In this paper, we have studied the problem of domain adaptation on graphs both theoretically and methodologically. We have first proposed a theoretical analysis of the performance of graph domain adaptation methods. We have considered a setting where a pair of graphs are constructed from data samples drawn from a source manifold and a target manifold. We have focused on a graph domain adaptation framework where the source and the target label functions are estimated such that they have similar spectra. We have proposed an upper bound on the estimation error of the target label function and studied its dependence on the number of data samples, the geometric properties of the data manifolds, and graph parameters such as edge weights and the number of neighbors of graph nodes. In particular, as far as the graph properties are concerned, our theoretical results suggest that a "balanced" graph topology improves the performance of learning where the numbers of neighbors are proportionate across different nodes, and too weak edge weights and edges between too distant samples are avoided. Based on these findings, we then proposed a graph domain adaptation algorithm that jointly learns the graph structures and the label functions. Experimental results on synthetic and real data sets suggest that the proposed method yields promising performance in problems concerning machine learning on graph domains. An interesting future direction may be the extension of the study to multiple domains.

References

[1] Fang Z, Zhang Z. Discriminative transfer learning on manifold. In: SIAM International Conference on Data Mining; Austin, TX, USA; 2013. pp. 539-547.

[2] Rohrbach M, Ebert S, Schiele B. Transfer learning in a transductive setting. In: Advances in Neural Information Processing Systems; Lake Tahoe, NV, USA; 2013. pp. 46-54.

[3] Wang C, Mahadevan S. Manifold alignment using procrustes analysis. In: International Conference on Machine Learning; Helsinki, Finland; 2008. pp. 1120-1127.

[4] Xiao M, Guo Y. Feature space independent semi-supervised domain adaptation via kernel matching. IEEE Transactions on Pattern Analysis and Machine Intelligence 2015. 37 (1): 54-66.

[5] Pilancı M, Vural E. Domain adaptation via transferring spectral properties of label functions on graphs. In: IEEE 12th Image, Video, and Multidimensional Signal Processing Workshop; Bordeaux, France; 2016. pp. 1-5.

[6] Huang J, Smola AJ, Gretton A, Borgwardt KM, Schölkopf B. Correcting sample selection bias by unlabeled data. In: Advances in Neural Information Processing Systems; Vancouver, BC, Canada; 2006. pp. 601-608.

[7] Khalighi S, Ribeiro B, Nunes U. Importance weighted import vector machine for unsupervised domain adaptation. IEEE Transactions on Cybernetics 2017; 47 (10): 3280-3292.

[8] Daumé III H, Kumar A, Saha A. Frustratingly easy semi-supervised domain adaptation. In: 2010 Workshop on Domain Adaptation for Natural Language Processing; Uppsala, Sweden; 2010. pp. 53-59.

[9] Duan L, Xu D, Tsang IW. Learning with augmented features for heterogeneous domain adaptation. In: International Conference on Machine Learning; Edinburgh, UK; 2012. pp. 1-8.

[10] Courty N, Flamary R, Tuia D, Rakotomamonjy A. Optimal transport for domain adaptation. IEEE Transactions on Pattern Analysis and Machine Intelligence 2017; 39 (9): 1853-1865.

[11] Fernando B, Habrard A, Sebban M, Tuytelaars T. Unsupervised visual domain adaptation using subspace alignment. In: IEEE International Conference on Computer Vision; Sydney, Australia; 2013. pp. 2960-2967.

[12] Gong B, Shi Y, Sha F, Grauman K. Geodesic flow kernel for unsupervised domain adaptation. In: IEEE Conference on Computer Vision and Pattern Recognition; Providence, RI, USA; 2012. pp. 2066-2073.

[13] Jiang M, Huang W, Huang Z, Yen GG. Integration of global and local metrics for domain adaptation learning via dimensionality reduction. IEEE Transactions on Cybernetics 2017; 47 (1): 38-51.

[14] Pan SJ, Tsang IW, Kwok JT, Yang Q. Domain adaptation via transfer component analysis. IEEE Transactions on Neural Networks 2011; 22 (2): 199-210.

[15] Yan H, Ding, Li P, Wang Q, Xu Y, Zuo W. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In: IEEE Conference on Computer Vision and Pattern Recognition; Honolulu, HI, USA; 2017. pp 945-954.

[16] Cheng L Pan SJ. Semi-supervised domain adaptation on manifolds. IEEE Transactions on Neural Networks and Learning Systems 2014; 25 (12): 2240-2249.

[17] Yao T, Pan Y, Ngo C, Li H, Mei T. Semi-supervised domain adaptation with subspace learning for visual recognition. In: IEEE Conference on Computer Vision and Pattern Recognition; Boston, MA, USA; 2015. pp. 2142-2150.

[18] Das D, Lee CSG. Sample-to-sample correspondence for unsupervised domain adaptation. Engineering Applications of Artificial Intelligence 2018; 73: 80-91.

[19] Eynard D, Kovnatsky A, Bronstein MM, Glashoff K, Bronstein AM. Multimodal manifold analysis by simultaneous diagonalization of Laplacians. IEEE Transactions on Pattern Analysis and Machine Intelligence 2015; 37 (12): 2505-2517.

[20] Rodolà E, Cosmo L, Bronstein MM, Torsello A, Cremers D. Partial functional correspondence. Computer Graphics Forum 2017; 36 (1): 222-236.

[21] Dong X, Thanou D, Frossard F, Vandergheynst P. Learning Laplacian matrix in smooth graph signal representations. IEEE Transactions on Signal Processing 2016; 64 (23): 6160-6173.

[22] Egilmez HE, Pavez E, Ortega A. Graph learning from data under Laplacian and structural constraints. IEEE Journal of Selected Topics in Signal Processing 2017; 11 (6): 825-841.

[23] Kalofolias V. How to learn a graph from smooth signals. In: International Conference on Artificial Intelligence and Statistics; Cadiz, Spain; 2016. pp. 920-929.

[24] Argyriou A, Herbster M, Pontil M. Combining graph Laplacians for semi-supervised learning. In: Advances in Neural Information Processing Systems; Vancouver, BC, Canada; 2005. pp. 67-74.

[25] Dai G, Yeung D. Kernel selection for semi-supervised kernel machines. In: International Conference on Machine Learning; Corvallis, OR, USA; 2007. pp. 185-192.

[26] Cortes C, Mansour Y, Mohri M. Learning bounds for importance weighting. In: Advances in Neural Information Processing Systems; Vancouver, BC, Canada; 2010. pp. 442-450.

[27] Ben-David S, Blitzer J, Crammer K, Kulesza A, Pereira F et al. A theory of learning from different domains. Machine Learning 2010; 79 (1-2): 151-175.

[28] Ben-David S, Blitzer J, Crammer K, Pereira F. Analysis of representations for domain adaptation. In: Advances in Neural Information Processing Systems; Vancouver, BC, Canada; 2006. pp. 137-144.

[29] Blitzer J, Crammer K, Kulesza A, Pereira F, Wortman J. Learning bounds for domain adaptation. In: Advances in Neural Information Processing Systems; Vancouver, BC, Canada; 2007. pp. 129-136.

[30] Mansour Y, Mohri M, Rostamizadeh A. Domain adaptation: Learning bounds and algorithms. In: 22nd Conference on Learning Theory; Montreal, QC, Canada; 2009. pp. 1-11.

[31] Chung FRK. Spectral Graph Theory. Providence, RI, USA: American Mathematical Society, 1997.

[32] Shuman DI, Narang SK, Frossard P, Ortega A, Vandergheynst P. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Processing Magazine 2013; 30 (3): 83-98.

[33] Hein M, Audibert JY, von Luxburg U. From graphs to manifolds - weak and strong pointwise consistency of graph Laplacians. In: 18th Annual Conference on Learning Theory; Bertinoro, Italy; 2005. pp. 470-485.

[34] Singer A. From graph to manifold Laplacian: The convergence rate. Applied and Computational Harmonic Analysis 2006; 21 (1): 128-134.

[35] Vural E. Domain adaptation on graphs by learning graph topologies: Theoretical analysis and an algorithm. arXiv Technical report, 2018.

[36] Wang C, Mahadevan S. Heterogeneous domain adaptation using manifold alignment. In: 22nd International Joint Conference on Artificial Intelligence; Barcelona, Spain; 2011. pp. 1541-1546.

[37] Nene SA, Nayar SK, Murase H. Columbia Object Image Library (COIL-20) Technical report No. CUCS-006-96. New York, NY, USA: Columbia University, 1996.