2016 23rd International Conference on Pattern Recognition (ICPR)
Cancún Center, Cancún, México, December 4-8, 2016

Constrained Local and Global Consistency for Semi-supervised Learning

Celso A. R. Sousa, Gustavo E. A. P. A. Batista
Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo, São Carlos, Brazil
email: { sousa, gbatista }@icmc.usp.br

***Abstract*—One of the widely used algorithms for graph-based semi-supervised learning (SSL) is the *Local and Global Consistency* (LGC). Such an algorithm can be viewed as a convex optimization problem that balances fitness on labeled examples and smoothness on the graph using a graph Laplacian. In this paper, we provide a novel graph-based SSL algorithm incorporating two normalization constraints into the regularization framework of LGC. We prove that our method has closed-form solution and generalizes two existing methods, being more flexible than the original ones. Through experiments on benchmark data sets, we show the effectiveness of our method, which consistently outperforms the competing methods.**

I. Introduction

In scenarios in which we deal with only a few labeled examples, *semi-supervised learning* (SSL) algorithms can be effective in comparison to purely supervised approaches. Among all SSL algorithms, graph-based methods have gained increased attention in the last few years [1], [2] partially due to their empirical effectiveness on benchmark data sets [3]. Most graph-based SSL algorithms are based on the optimization of a *convex* cost function that uses a Laplacian regularizer term as smoothness functional. Such an optimization problem is possibly subject to some fitting and/or normalization constraints. Formally, the Laplacian regularizer term is a smoothness penalty term that tries to reflect the intrinsic geometric structure of the data marginal distribution.

A widely used method for graph-based SSL is the *Local and Global Consistency* (LGC) [4], which achieves state-of-the-art classification performance with respect to graph construction and parameter selection on several data sets [3], [5]. LGC can be formulated as a *convex* optimization problem that balances fitness on labeled examples and smoothness on the graph through a Laplacian regularizer term.

Another state-of-the-art graph-based SSL algorithm is the *Robust Multi-class Graph Transduction* (RMGT) [6], which incorporates two normalization constraints into the regularization framework of the *Gaussian Fields and Harmonic Functions* (GFHF) [7] algorithm. RMGT has been generalized for any *positive semidefinite* (PSD) matrix in [8]. Such a method is called *Robust Multi-class Graph Transduction with Higher Order Regularization* (RMGTHOR) and can naturally deal with a variety of graph Laplacians. Moreover, RMGTHOR consistently outperforms other state-of-the-art graph-based SSL algorithms [8].

In this paper, we focus on the problem of *graph transduction*. Consequently, we want to classify the unlabeled examples using a weighted graph generated from the training sample and (scarce) label information without the necessity to provide generalization for the entire sample space. Specifically, we provide in this paper a novel graph-based SSL algorithm based on LGC, generalizing RMGT and RMGTHOR.

A. Motivation

Recent experimental evaluations [5], [9], [3] show the effectiveness of RMGT and RMGTHOR over state-of-the-art graph-based SSL algorithms on benchmark data sets with respect to graph construction and parameter selection. In many experimental settings, RMGT and RMGTHOR outperformed the competing methods by a large margin [9].

RMGT was specifically designed for the combinatorial Laplacian, which may not be the most appropriate graph Laplacian for a given application [10]. Our method can naturally deal with a variety of graph Laplacians, being more flexible than RMGT. Moreover, we can also apply *higher order regularization* in our method, which can be effective on general SSL tasks [11]. Therefore, we expect that our method achieves better classification performance than RMGT in general SSL tasks.

In [3], the authors showed that LGC consistently outperforms GFHF. Since RMGT and RMGTHOR are based on GFHF and our method is based on LGC, we expect that our method achieves slightly better classification performance than RMGT and RMGTHOR.

B. Contributions

The contributions of this paper are summarized as follows:

- we provide a novel graph-based SSL algorithm, based on LGC, called *Constrained Local and Global Consistency* (CLGC);
- we show that our method has closed-form solution;
- we prove that our method generalizes RMGT and RMGTHOR;
- we show the effectiveness of our method on benchmark data sets against state-of-the-art graph-based SSL algorithms.

978-1-5090-4846-5/16/$31.00 ©2016 IEEE

C. Outline

The remainder of this paper is organized as follows. Section II provides a background on graph-based SSL. Section III formulates our method. Section IV provides our experimental evaluation. Finally, Section V describes our conclusions.

II. Background

Consider a training sample $\mathcal{X} := \{\boldsymbol{x}_i\}_{i=1}^n \subset \mathbb{R}^d$, in which the first l (usually $l \ll n$) examples are labeled with one of c classes; the remainder $u := n - l$ examples are unlabeled. Let $\mathbb{B} := \{0, 1\}$ and $\mathbb{N}_a := \{i \in \mathbb{N}^* | 1 \le i \le a\}$, $\forall a \in \mathbb{N}^*$. Assume that $\boldsymbol{x}_i$, $i \in \mathbb{N}_l$, has label $y_i \in \mathbb{N}_c$. Let $\mathcal{N}_i \subset \mathcal{X}$ be the set of neighbors of $\boldsymbol{x}_i$ and $\boldsymbol{x}_i^{(k)} \in \mathcal{X}$ the k-th nearest neighbor of $\boldsymbol{x}_i$. Let $\mathbf{Y} \in \mathbb{B}^{l \times c}$ be a label matrix in which $\mathbf{Y}_{ij} = 1$ if and only if $\boldsymbol{x}_i$ has label $y_i = j$. Consider $\mathbf{1}_n$ and $\mathbf{0}_n$ n-dimensional 1-entry and 0-entry vectors, respectively. Let $\mathbf{A}_{i,\cdot} \in \mathbb{R}^{1 \times b}$ and $\mathbf{A}_{\cdot,j} \in \mathbb{R}^a$ be the i-th row and j-th column vectors of a matrix $\mathbf{A} \in \mathbb{R}^{a \times b}$, $\forall a, b \in \mathbb{N}^*$.

Let $\mathbf{W} \in \mathbb{R}^{n \times n}$ be a weighted matrix generated from $\mathcal{X}$ and $\mathbf{F} \in \mathbb{R}^{n \times c}$ be the output of a given graph-based SSL algorithm. Assume that $\mathbf{F}$ and $\mathbf{Y}$ are subdivided into two submatrices while all other matrices are subdivided into four submatrices. For instance:

$$\mathbf{W} := \begin{bmatrix} \mathbf{W}_{\mathcal{LL}} & \mathbf{W}_{\mathcal{LU}} \\ \mathbf{W}_{\mathcal{UL}} & \mathbf{W}_{\mathcal{UU}} \end{bmatrix} \qquad \mathbf{Y} := \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ \mathbf{Y}_{\mathcal{U}} \end{bmatrix}$$

where $\mathbf{W}_{\mathcal{LL}} \in \mathbb{R}^{l \times l}$ and $\mathbf{Y}_{\mathcal{L}} \in \mathbb{R}^{l \times c}$ are the submatrices of $\mathbf{W}$ and $\mathbf{Y}$, respectively, on labeled examples, $\mathbf{W}_{\mathcal{UU}} \in \mathbb{R}^{u \times u}$ and $\mathbf{Y}_{\mathcal{U}} \in \mathbb{R}^{u \times c}$ are the submatrices of $\mathbf{W}$ and $\mathbf{Y}$, respectively, on unlabeled examples, and so on. By definition, $\mathbf{Y}_{\mathcal{U}} = \mathbf{O}_{u \times c}$ in which $\mathbf{O}_{u \times c}$ is the u-by-c null matrix. Since we focus on multi-class problems, we have $\mathbf{Y}_{\mathcal{L}} \mathbf{1}_c = \mathbf{1}_l$.

Let $\mathbf{L} \in \mathbb{R}^{n \times n}$ be a graph Laplacian generated from $\mathbf{W}$ and $\mathbf{I}_n$ the n-by-n identity matrix. For instance, the *unnormalized* Laplacian is defined by $\mathbf{L}_{\mathbb{U}} := \mathbf{D} - \mathbf{W}$ where $\mathbf{D} := \operatorname{diag}(\mathbf{W}\mathbf{1}_n)$ and the *normalized* Laplacian is defined by $\mathbf{L}_{\mathbb{N}} := \mathbf{I}_n - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$. Although $\mathbf{L}_{\mathbb{U}}$ and $\mathbf{L}_{\mathbb{N}}$ are the most commonly used graph Laplacians in the SSL literature [2], [1], there are other graph Laplacians that can also be applied in SSL as well [12], [13]. For instance, the *iterated* Laplacian is defined by $\mathbf{L}_{\mathbb{I}} := \mathbf{L}^p$ in which $p \in \mathbb{N}^*$ is the Laplacian's degree.

A. Unconstrained graph-based SSL

Given a graph Laplacian $\mathbf{L}$, we are able to apply graph-based SSL algorithms to perform classification. The GFHF algorithm[1] [7] can be viewed as the following optimization problem:

$$\min_{\mathbf{F} \in \mathbb{R}^{n \times c}} \operatorname{tr}\left(\mathbf{F}^\top \mathbf{L} \mathbf{F}\right) \quad \text{s.t.} \quad \mathbf{F}_{\mathcal{L}} = \mathbf{Y}_{\mathcal{L}}$$

where $\operatorname{tr}(\cdot)$ is the trace of a matrix. Since $\mathbf{L} \succcurlyeq 0$, we obtain the following closed-form solution:

$$\mathbf{F} = \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{UU}}^{-1} \mathbf{L}_{\mathcal{UL}} \mathbf{Y}_{\mathcal{L}} \end{bmatrix} \tag{1}$$

The LGC algorithm[2] [4] can be viewed as the following optimization problem:

$$\min_{\mathbf{F} \in \mathbb{R}^{n \times c}} \operatorname{tr}\left(\mathbf{F}^\top \mathbf{L} \mathbf{F} + (\mathbf{F} - \mathbf{Y})^\top \boldsymbol{\Sigma} (\mathbf{F} - \mathbf{Y})\right) \tag{2}$$

where $\boldsymbol{\Sigma} := \begin{bmatrix} \boldsymbol{\Sigma}_{\mathcal{L}} & \mathbf{O}_{l \times u} \\ \mathbf{O}_{u \times l} & \boldsymbol{\Sigma}_{\mathcal{U}} \end{bmatrix} \in \mathbb{R}^{n \times n}$ is a diagonal matrix that contains the regularization parameters. In the original formulation, we have $\boldsymbol{\Sigma} = \mu \mathbf{I}_n$ where $\mu \in \mathbb{R}_+^*$ is the "global" regularization parameter. Since $\mathbf{L} \succcurlyeq 0$, we obtain the following closed-form solution:

$$\mathbf{F} = (\mathbf{L} + \boldsymbol{\Sigma})^{-1} \boldsymbol{\Sigma} \mathbf{Y} \tag{3}$$

Proposition 1 ([14]): If $\boldsymbol{\Sigma}_{\mathcal{U}} = \mathbf{O}_{u \times u}$ and $\boldsymbol{\Sigma}_{\mathcal{L}}^{-1} \to \mathbf{O}_{l \times l}$, (3) *is reduced to* (1).

B. Constrained graph-based SSL

In [6], the authors proposed the following two normalization constraints for graph-based SSL: (1) $\mathbf{F}\mathbf{1}_c = \mathbf{1}_n$; and (2) $\mathbf{F}^\top \mathbf{1}_n = n\boldsymbol{\omega}$ where $\boldsymbol{\omega} \in \mathbb{R}^c$ can be the class prior probabilities or the uniform class distribution ($\boldsymbol{\omega} = \mathbf{1}_c / c$). These constraints were incorporated into the GFHF algorithm for $\mathbf{L} = \mathbf{L}_{\mathbb{U}}$. Mathematically, the RMGT algorithm [6] can be viewed as the following constrained optimization problem:

$$\begin{aligned} &\min_{\mathbf{F} \in \mathbb{R}^{n \times c}} \operatorname{tr}\left(\mathbf{F}^\top \mathbf{L}_{\mathbb{U}} \mathbf{F}\right) \\ &\text{s.t.} \quad \mathbf{F}_{\mathcal{L}} = \mathbf{Y}_{\mathcal{L}}, \;\; \mathbf{F}\mathbf{1}_c = \mathbf{1}_n, \;\; \mathbf{F}^\top \mathbf{1}_n = n\boldsymbol{\omega} \end{aligned} \tag{4}$$

The closed-form solution of (4) is given by:

$$\mathbf{F} = \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -(\mathbf{L}_{\mathbb{U}})_{\mathcal{UU}}^{-1} (\mathbf{L}_{\mathbb{U}})_{\mathcal{UL}} \mathbf{Y}_{\mathcal{L}} + \dfrac{(\mathbf{L}_{\mathbb{U}})_{\mathcal{UU}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top (\mathbf{L}_{\mathbb{U}})_{\mathcal{UU}}^{-1} \mathbf{1}_u} \boldsymbol{\zeta} \end{bmatrix} \tag{5}$$

in which

$$\boldsymbol{\zeta} = n\boldsymbol{\omega}^\top - \mathbf{1}_l^\top \mathbf{Y}_{\mathcal{L}} + \mathbf{1}_u^\top (\mathbf{L}_{\mathbb{U}})_{\mathcal{UU}}^{-1} (\mathbf{L}_{\mathbb{U}})_{\mathcal{UL}} \mathbf{Y}_{\mathcal{L}}$$

The RMGTHOR algorithm [8] is a generalization of (4) for any graph Laplacian. The closed-form solution of RMGTHOR is given by:

$$\mathbf{F} = \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{UU}}^{-1} \mathbf{L}_{\mathcal{UL}} \mathbf{Y}_{\mathcal{L}} + \dfrac{\mathbf{L}_{\mathcal{UU}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{UU}}^{-1} \mathbf{1}_u} \boldsymbol{\zeta} + \dfrac{1}{c} \boldsymbol{\nu} \mathbf{1}_c^\top \end{bmatrix} \tag{6}$$

in which

[1] See [14] for a nice review on the GFHF algorithm.

[2] See [15] for a nice review on the LGC algorithm.

$$\boldsymbol{\zeta} = n\boldsymbol{\omega}^\top - \mathbf{1}_l^\top \mathbf{Y}_\mathcal{L} + \mathbf{1}_u^\top \mathbf{L}_{\mathcal{UU}}^{-1}\mathbf{L}_{\mathcal{UL}}\mathbf{Y}_\mathcal{L}$$
$$\boldsymbol{\nu} = \mathbf{1}_u + \mathbf{L}_{\mathcal{UU}}^{-1}\mathbf{L}_{\mathcal{UL}}\mathbf{1}_l - \frac{\mathbf{L}_{\mathcal{UU}}^{-1}\mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{UU}}^{-1}\mathbf{1}_u}\left(u + \mathbf{1}_u^\top \mathbf{L}_{\mathcal{UU}}^{-1}\mathbf{L}_{\mathcal{UL}}\mathbf{1}_l\right)$$

Corollary 1 ([8]): If $\mathbf{L} = \mathbf{L}_\mathbb{U}$, (6) *is reduced to* (5).

III. Proposed method

This section formulates our method, called CLGC, and highlights its strenghts and weaknesses based on theoretical and experimental results in previous research [4], [8], [9].

A. Regularization framework

Our method is based on the regularization framework in (2), incorporating the normalization constraints in [6]. Mathematically, our method is formulated as the following optimization problem:

$$\begin{aligned} \min_{\mathbf{F}\in\mathbb{R}^{n\times c}} & \operatorname{tr}\left(\mathbf{F}^\top\mathbf{L}\mathbf{F} + (\mathbf{F}-\mathbf{Y})^\top\boldsymbol{\Sigma}(\mathbf{F}-\mathbf{Y})\right) \\ \text{s.t.} & \ \mathbf{F}\mathbf{1}_c = \mathbf{1}_n, \ \ \mathbf{F}^\top\mathbf{1}_n = n\boldsymbol{\omega} \end{aligned} \tag{7}$$

Proposition 2: The closed-form solution of (7) *is given by*

$$\mathbf{F} = \boldsymbol{\Theta}\boldsymbol{\Sigma}\mathbf{Y} + \frac{\boldsymbol{\Theta}\mathbf{1}_n}{\mathbf{1}_n^\top\boldsymbol{\Theta}\mathbf{1}_n}\boldsymbol{\zeta} + \frac{1}{c}\boldsymbol{\nu}\mathbf{1}_c^\top \tag{8}$$

where

$$\boldsymbol{\Theta} = (\mathbf{L}+\boldsymbol{\Sigma})^{-1}$$
$$\boldsymbol{\zeta} = n\boldsymbol{\omega}^\top - \mathbf{1}_n^\top\boldsymbol{\Theta}\boldsymbol{\Sigma}\mathbf{Y}$$
$$\boldsymbol{\nu} = \mathbf{1}_n - \boldsymbol{\Theta}\boldsymbol{\Sigma}\mathbf{Y}\mathbf{1}_c - \frac{\boldsymbol{\Theta}\mathbf{1}_n}{\mathbf{1}_n^\top\boldsymbol{\Theta}\mathbf{1}_n}\left[n - \mathbf{1}_n^\top\boldsymbol{\Theta}\boldsymbol{\Sigma}\mathbf{Y}\mathbf{1}_c\right]$$

Proof: The Lagrangian corresponding to (7) is given by

$$\begin{aligned} \mathcal{L}(\mathbf{F},\boldsymbol{\xi},\boldsymbol{\lambda}) = & \operatorname{tr}\left(\mathbf{F}^\top\mathbf{L}\mathbf{F} + (\mathbf{F}-\mathbf{Y})^\top\boldsymbol{\Sigma}(\mathbf{F}-\mathbf{Y})\right) \\ & - \boldsymbol{\xi}^\top(\mathbf{F}\mathbf{1}_c - \mathbf{1}_n) - \boldsymbol{\lambda}^\top\left(\mathbf{F}^\top\mathbf{1}_n - n\boldsymbol{\omega}\right) \end{aligned}$$

where $\boldsymbol{\xi} \in \mathbb{R}^n$ and $\boldsymbol{\lambda} \in \mathbb{R}^c$ are the Lagrange multipliers. Zeroing $\partial\mathcal{L}/\partial\mathbf{F}$, we obtain:

$$\mathbf{F} = \boldsymbol{\Theta}\left(\boldsymbol{\Sigma}\mathbf{Y} + \frac{1}{2}\boldsymbol{\xi}\mathbf{1}_c^\top + \frac{1}{2}\mathbf{1}_n\boldsymbol{\lambda}^\top\right) \tag{9}$$

where $\boldsymbol{\Theta} = (\mathbf{L}+\boldsymbol{\Sigma})^{-1}$. Substituting (9) in the constraint $\mathbf{F}\mathbf{1}_c = \mathbf{1}_n$, we obtain:

$$\boldsymbol{\Theta}\boldsymbol{\Sigma}\mathbf{Y}\mathbf{1}_c + \frac{c}{2}\boldsymbol{\Theta}\boldsymbol{\xi} + \frac{1}{2}\boldsymbol{\Theta}\mathbf{1}_n\boldsymbol{\lambda}^\top\mathbf{1}_c = \mathbf{1}_n \tag{10}$$

Substituting (9) in the constraint $\mathbf{F}^\top\mathbf{1}_n = n\boldsymbol{\omega}$, we obtain:

$$\boldsymbol{\lambda}^\top = \frac{2}{\mathbf{1}_n^\top\boldsymbol{\Theta}\mathbf{1}_n}\boldsymbol{\lambda}^{(0)} \tag{11}$$

in which

$$\boldsymbol{\lambda}^{(0)} = n\boldsymbol{\omega}^\top - \mathbf{1}_n^\top\boldsymbol{\Theta}\boldsymbol{\Sigma}\mathbf{Y} - \frac{1}{2}\mathbf{1}_n^\top\boldsymbol{\Theta}\boldsymbol{\xi}\mathbf{1}_c^\top$$

Substituting (11) in (10), we obtain:

$$\boldsymbol{\xi} = \frac{2}{c}\left(\boldsymbol{\Theta} - \frac{\boldsymbol{\Theta}\mathbf{1}_n\mathbf{1}_n^\top\boldsymbol{\Theta}}{\mathbf{1}_n^\top\boldsymbol{\Theta}\mathbf{1}_n}\right)^{-1}\left(\mathbf{1}_n - \boldsymbol{\Theta}\boldsymbol{\Sigma}\mathbf{Y}\mathbf{1}_c - \boldsymbol{\xi}^{(0)}\right) \tag{12}$$

in which

$$\boldsymbol{\xi}^{(0)} = \frac{\boldsymbol{\Theta}\mathbf{1}_n}{\mathbf{1}_n^\top\boldsymbol{\Theta}\mathbf{1}_n}\left(n - \mathbf{1}_n^\top\boldsymbol{\Theta}\boldsymbol{\Sigma}\mathbf{Y}\mathbf{1}_c\right)$$

Substituting (11) in (9), we obtain:

$$\mathbf{F} = \boldsymbol{\Theta}\boldsymbol{\Sigma}\mathbf{Y} + \frac{\boldsymbol{\Theta}\mathbf{1}_n}{\mathbf{1}_n^\top\boldsymbol{\Theta}\mathbf{1}_n}\boldsymbol{\zeta} + \frac{1}{2}\boldsymbol{\varpi}\boldsymbol{\xi}\mathbf{1}_c^\top \tag{13}$$

in which

$$\boldsymbol{\zeta} = n\boldsymbol{\omega}^\top - \mathbf{1}_n^\top\boldsymbol{\Theta}\boldsymbol{\Sigma}\mathbf{Y}$$
$$\boldsymbol{\varpi} = \boldsymbol{\Theta}\left[\mathbf{I}_n - \frac{\mathbf{1}_n\mathbf{1}_n^\top\boldsymbol{\Theta}}{\mathbf{1}_n^\top\boldsymbol{\Theta}\mathbf{1}_n}\right]$$

Substituting (12) in (13), we obtain (8). ∎

B. Special cases

In this section, we provide special cases of our method. Specifically, Proposition 3 shows that our method generalizes RMGTHOR and RMGT. Corollary 2 shows that the constraint $\mathbf{F}\mathbf{1}_c = \mathbf{1}_n$ is always satisfied under certain conditions. Therefore, such a constraint can be dropped to the original optimization problem (under conditions in Corollary 2) without changing the closed-form solution.

Proposition 3: If $\boldsymbol{\Sigma}_\mathcal{U} = \mathbf{O}_{u\times u}$ and $\boldsymbol{\Sigma}_\mathcal{L}^{-1} \to \mathbf{O}_{l\times l}$, (8) *is reduced to* (6). *If we also impose that $\mathbf{L} = \mathbf{L}_\mathbb{U}$,* (8) *is reduced to* (5).

Proof: Let $\boldsymbol{\Theta} = (\mathbf{L}+\boldsymbol{\Sigma})^{-1}$. If $\boldsymbol{\Sigma}_\mathcal{U} = \mathbf{O}_{u\times u}$, we can write:

$$\begin{aligned} \begin{bmatrix} \mathbf{L}_{\mathcal{LL}} + \boldsymbol{\Sigma}_\mathcal{L} & \mathbf{L}_{\mathcal{LU}} \\ \mathbf{L}_{\mathcal{UL}} & \mathbf{L}_{\mathcal{UU}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Theta}_{\mathcal{LL}} & \boldsymbol{\Theta}_{\mathcal{LU}} \\ \boldsymbol{\Theta}_{\mathcal{UL}} & \boldsymbol{\Theta}_{\mathcal{UU}} \end{bmatrix} = \\ = \begin{bmatrix} \mathbf{I}_l & \mathbf{O}_{l\times u} \\ \mathbf{O}_{u\times l} & \mathbf{I}_u \end{bmatrix} \end{aligned}$$

This yields:

$$\begin{aligned} \boldsymbol{\Theta}_{\mathcal{LL}} &= \boldsymbol{\Gamma}^{-1}\boldsymbol{\Sigma}_\mathcal{L}^{-1} \\ \boldsymbol{\Theta}_{\mathcal{UL}} &= -\mathbf{L}_{\mathcal{UU}}^{-1}\mathbf{L}_{\mathcal{UL}}\boldsymbol{\Gamma}^{-1}\boldsymbol{\Sigma}_\mathcal{L}^{-1} \\ \boldsymbol{\Theta}_{\mathcal{LU}} &= -\boldsymbol{\Gamma}^{-1}\boldsymbol{\Sigma}_\mathcal{L}^{-1}\mathbf{L}_{\mathcal{LU}}\mathbf{L}_{\mathcal{UU}}^{-1} \\ \boldsymbol{\Theta}_{\mathcal{UU}} &= \mathbf{L}_{\mathcal{UU}}^{-1} + \mathbf{L}_{\mathcal{UU}}^{-1}\mathbf{L}_{\mathcal{UL}}\boldsymbol{\Gamma}^{-1}\boldsymbol{\Sigma}_\mathcal{L}^{-1}\mathbf{L}_{\mathcal{LU}}\mathbf{L}_{\mathcal{UU}}^{-1} \end{aligned} \tag{14}$$

in which

$$\boldsymbol{\Gamma} = \boldsymbol{\Sigma}_{\mathcal{L}}^{-1}\mathbf{L}_{\mathcal{LL}} + \mathbf{I}_l - \boldsymbol{\Sigma}_{\mathcal{L}}^{-1}\mathbf{L}_{\mathcal{LU}}\mathbf{L}_{\mathcal{UU}}^{-1}\mathbf{L}_{\mathcal{UL}}$$

Therefore, Eq. (8) can be rewritten as:

$$\begin{aligned} \mathbf{F} = &\begin{bmatrix} \boldsymbol{\Theta}_{\mathcal{LL}}\boldsymbol{\Sigma}_{\mathcal{L}}\mathbf{Y}_{\mathcal{L}} \\ \boldsymbol{\Theta}_{\mathcal{UL}}\boldsymbol{\Sigma}_{\mathcal{L}}\mathbf{Y}_{\mathcal{L}} \end{bmatrix} + \boldsymbol{\beta}^{(0)}\boldsymbol{\beta}^{(1)} \\ &+ \frac{1}{c}\left\{ \begin{bmatrix} \mathbf{1}_l - \boldsymbol{\Theta}_{\mathcal{LL}}\boldsymbol{\Sigma}_{\mathcal{L}}\mathbf{1}_l \\ \mathbf{1}_u - \boldsymbol{\Theta}_{\mathcal{UL}}\boldsymbol{\Sigma}_{\mathcal{L}}\mathbf{1}_l \end{bmatrix} - \boldsymbol{\beta}^{(0)}\boldsymbol{\beta}^{(2)} \right\} \mathbf{1}_c^\top \end{aligned} \tag{15}$$

in which

$$\begin{aligned} \boldsymbol{\beta}^{(0)} &= \frac{1}{\tau}\begin{bmatrix} \boldsymbol{\Theta}_{\mathcal{LL}}\mathbf{1}_l + \boldsymbol{\Theta}_{\mathcal{LU}}\mathbf{1}_u \\ \boldsymbol{\Theta}_{\mathcal{UL}}\mathbf{1}_l + \boldsymbol{\Theta}_{\mathcal{UU}}\mathbf{1}_u \end{bmatrix} \\ \boldsymbol{\beta}^{(1)} &= n\boldsymbol{\omega}^\top - \mathbf{1}_l^\top\boldsymbol{\Theta}_{\mathcal{LL}}\boldsymbol{\Sigma}_{\mathcal{L}}\mathbf{Y}_{\mathcal{L}} - \mathbf{1}_u^\top\boldsymbol{\Theta}_{\mathcal{UL}}\boldsymbol{\Sigma}_{\mathcal{L}}\mathbf{Y}_{\mathcal{L}} \\ \boldsymbol{\beta}^{(2)} &= n - \mathbf{1}_l^\top\boldsymbol{\Theta}_{\mathcal{LL}}\boldsymbol{\Sigma}_{\mathcal{L}}\mathbf{1}_l - \mathbf{1}_u^\top\boldsymbol{\Theta}_{\mathcal{UL}}\boldsymbol{\Sigma}_{\mathcal{L}}\mathbf{1}_l \end{aligned} \tag{16}$$

such that

$$\begin{aligned} \tau = &\mathbf{1}_l^\top\boldsymbol{\Theta}_{\mathcal{LL}}\mathbf{1}_l + \mathbf{1}_u^\top\boldsymbol{\Theta}_{\mathcal{UL}}\mathbf{1}_l + \mathbf{1}_l^\top\boldsymbol{\Theta}_{\mathcal{LU}}\mathbf{1}_u \\ &+ \mathbf{1}_u^\top\boldsymbol{\Theta}_{\mathcal{UU}}\mathbf{1}_u \end{aligned}$$

If $\boldsymbol{\Sigma}_{\mathcal{L}}^{-1} \to \mathbf{O}_{l\times l}$, then $\boldsymbol{\Gamma} \to \mathbf{I}_l$. Therefore, we have:

$$\begin{aligned} &\boldsymbol{\Theta}_{\mathcal{LU}} \approx \mathbf{O}_{l\times u}, \quad \boldsymbol{\Theta}_{\mathcal{UU}} \approx \mathbf{L}_{\mathcal{UU}}^{-1}, \quad \boldsymbol{\Theta}_{\mathcal{LL}}\boldsymbol{\Sigma}_{\mathcal{L}} \approx \mathbf{I}_l, \\ &\boldsymbol{\Theta}_{\mathcal{LL}} \approx \mathbf{O}_{l\times l}, \quad \boldsymbol{\Theta}_{\mathcal{UL}}\boldsymbol{\Sigma}_{\mathcal{L}} = -\mathbf{L}_{\mathcal{UU}}^{-1}\mathbf{L}_{\mathcal{UL}} \\ &\boldsymbol{\Theta}_{\mathcal{UL}} \approx \mathbf{O}_{u\times l} \end{aligned} \tag{17}$$

Substituting the approximations in (17) in (16), we obtain:

$$\begin{aligned} \boldsymbol{\beta}^{(0)} &\approx \frac{1}{\mathbf{1}_u^\top\mathbf{L}_{\mathcal{UU}}^{-1}\mathbf{1}_u}\begin{bmatrix} \mathbf{0}_l \\ \mathbf{L}_{\mathcal{UU}}^{-1}\mathbf{1}_u \end{bmatrix} \\ \boldsymbol{\beta}^{(1)} &\approx n\boldsymbol{\omega}^\top - \mathbf{1}_l^\top\mathbf{Y}_{\mathcal{L}} + \mathbf{1}_u^\top\mathbf{L}_{\mathcal{UU}}^{-1}\mathbf{L}_{\mathcal{UL}}\mathbf{Y}_{\mathcal{L}} \\ \boldsymbol{\beta}^{(2)} &\approx u + \mathbf{1}_u^\top\mathbf{L}_{\mathcal{UU}}^{-1}\mathbf{L}_{\mathcal{UL}}\mathbf{1}_l \end{aligned} \tag{18}$$

Substituting $\boldsymbol{\beta}^{(0)}$, $\boldsymbol{\beta}^{(1)}$, and $\boldsymbol{\beta}^{(2)}$ in (15), we obtain (6). From Corollary 1, if $\mathbf{L} = \mathbf{L}_{\mathbb{U}}$ in (6), we obtain (5). ■

Corollary 2: If $\boldsymbol{\Sigma}_{\mathcal{U}} = \mathbf{O}_{u\times u}$, $\boldsymbol{\Sigma}_{\mathcal{L}}^{-1} \to \mathbf{O}_{l\times l}$, and $\mathbf{L} = \mathbf{L}_{\mathbb{U}}$, the closed-form solution of (7) *is equivalent to the closed-form solution of the following optimization problem:*

$$\begin{aligned} \min_{\mathbf{F}\in\mathbb{R}^{n\times c}} \; &\operatorname{tr}\left(\mathbf{F}^\top\mathbf{L}_{\mathbb{U}}\mathbf{F} + (\mathbf{F}-\mathbf{Y})^\top\boldsymbol{\Sigma}(\mathbf{F}-\mathbf{Y})\right) \\ &\text{s.t.} \;\; \mathbf{F}^\top\mathbf{1}_n = n\boldsymbol{\omega} \end{aligned} \tag{19}$$

Proof: The proof can be done by following the steps in Proposition 2. ■

C. Strenghts and weaknesses

Our method has the following strenghts:

- it has closed-form solution and is easy to implement;
- it generalizes two existing graph-based SSL algorithms (RMGTHOR and RMGT), being more flexible than its predecessors;
- it achieves state-of-the-art classification performance.

Our method has the following weaknesses:

- it runs in $\mathcal{O}\left(n^3\right)$ time, which is the same time complexity of widely used graph-based SSL algorithms [6], [7], [4], [16]. This may be unfeasible even for data sets with moderate size. Probably, this issue might be solved by applying in our method recent approaches that scale up graph-based SSL [17], [18];
- since our method generalizes the RMGT and RMGTHOR algorithms, it may inherit their weaknesses. Specifically, since the RMGT and RMGTHOR algorithms may not be effective on data sets with *high*[3] unbalanced ratio [8], [3], [5], our method might also be uneffective in this scenario;
- as opposed to the LGC algorithm, we do not have an iterative method to solve (7) with convergence guarantees. Such a method is still under development.

IV. Experimental evaluation

In this section, we empirically evaluate our method against six state-of-the-art graph-based SSL algorithms. We used six benchmark data sets in [1][4]: USPS; COIL_2; DIGIT-1; G-241N; G-241C; and TEXT. We used the preprocessing described in [3][5]. Due to reasons concerning reproducibility, the source code used in our experiments as well as our experimental results are freely available[6].

A. Experimental setup

We used the experimental protocol in [3] to compare our results with those in [3], [8]. Specifically, we empirically compare our method against the following graph-based SSL algorithms[7]: GFHF [7]; LGC [4]; *Laplacian Regularized Least Squares* (LapRLS) [16]; *Laplacian Support Vector Machine* (LapSVM) [16]; RMGT [6]; and RMGTHOR [8].

In order to compute a distance matrix $\boldsymbol{\Psi} \in \mathbb{R}^{n\times n}$ from $\mathcal{X}$, we chose the cosine distance for the TEXT data set and the L_2-norm for the other data sets. From $\boldsymbol{\Psi}$, we generated an adjacency matrix $\mathbf{A} \in \mathbb{B}^{n\times n}$ using *symmetric k-nearest neighbors* (symKNN), which creates an edge between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ if $\boldsymbol{x}_j$ is one of the k closest examples of $\boldsymbol{x}_i$ or vice versa. The values of k were chosen in the set $\{4, 6, 8, \cdots, 40\}$.

[3]The results in [5] are related to data sets in which the majority class has at least two times more examples than the minority class.

[4]http://olivier.chapelle.cc/ssl-book/benchmarks.html.

[5]http://sites.labic.icmc.usp.br/sousa/experiments_graph_SSL/.

[6]http://sites.labic.icmc.usp.br/sousa/constrained_graph_SSL/.

[7]Extensive results and parameter settings for GFHF, LGC, LapRLS, LapSVM, RMGT, and RMGTHOR can be found in [3], [19], [8].

TABLE I. BEST AVERAGE ERROR RATES (%) AND STANDARD DEVIATIONS (%) FOR THE SSL ALGORITHMS USING THE SYMKNN-RBF GRAPH.

$l = 10$	USPS	$COIL_2$	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	11.07 (3.33)	35.13 (6.92)	10.19 (4.27)	46.12 (7.60)	46.27 (6.98)	39.15 (5.69)	5.667
LGC	11.22 (3.07)	34.96 (6.69)	10.68 (4.91)	38.06 (6.91)	40.24 (5.13)	35.42 (5.58)	4.667
LapRLS	**10.99 (3.05)**	34.92 (5.98)	10.22 (4.25)	38.09 (6.76)	40.35 (6.23)	35.12 (5.68)	4.0
LapSVM	11.42 (4.03)	34.95 (6.81)	9.42 (3.97)	39.15 (6.07)	40.91 (6.08)	39.88 (6.01)	5.167
RMGT	16.62 (2.90)	31.05 (4.80)	8.63 (3.35)	44.99 (6.97)	38.44 (6.22)	30.42 (6.26)	4.167
RMGTHOR	13.32 (3.64)	30.98 (3.52)	7.42 (3.37)	**36.11 (16.00)**	26.94 (4.95)	30.22 (2.47)	2.5
CLGC	12.50 (3.23)	**30.93 (5.34)**	**5.72 (2.57)**	36.13 (15.98)	**26.85 (5.06)**	**28.89 (1.94)**	**1.833**

$l = 100$	USPS	$COIL_2$	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	3.30 (0.97)	1.14 (1.46)	2.22 (0.52)	32.38 (5.69)	37.89 (5.60)	24.79 (2.89)	5.333
LGC	3.56 (1.00)	1.17 (1.48)	2.46 (0.65)	22.38 (1.58)	29.98 (2.91)	24.26 (2.18)	5.167
LapRLS	3.29 (1.01)	**0.80 (1.13)**	2.17 (0.43)	22.14 (1.59)	28.73 (3.43)	24.36 (1.58)	3.583
LapSVM	4.10 (1.45)	**0.80 (1.14)**	2.18 (0.47)	21.64 (3.90)	25.51 (5.58)	23.74 (2.10)	3.583
RMGT	6.56 (2.24)	1.92 (1.74)	2.37 (0.43)	28.21 (8.55)	25.87 (2.39)	21.00 (1.24)	5.0
RMGTHOR	3.80 (1.05)	3.09 (1.79)	2.14 (0.40)	9.95 (0.87)	18.56 (0.93)	21.17 (0.92)	3.5
CLGC	**3.17 (0.57)**	2.79 (1.22)	**1.99 (0.51)**	**9.90 (0.97)**	**17.18 (0.90)**	**20.68 (1.07)**	**1.833**

In order to generate $\mathbf{W}$ from $\mathbf{A}$ and $\mathbf{\Psi}$, we used the RBF kernel[8], which is defined by:

$$\mathcal{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\Psi^2(\boldsymbol{x}_i, \boldsymbol{x}_j)}{2\sigma^2}\right)$$

such that $\sigma \in \mathbb{R}_+^*$ is the kernel's bandwidth parameter and $\Psi : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$ is a distance function. For all data sets, we estimate the value of the bandwidth parameter σ in the RBF kernel by $\sigma = \frac{1}{3n}\sum_{i=1}^{n} \Psi\left(\boldsymbol{x}_i, \boldsymbol{x}_i^{(k)}\right)$, as suggested in [20].

From $\mathbf{W}$, we generated the unnormalized Laplacian $\mathbf{L}_{\mathbb{U}}$ as $\mathbf{L}_{\mathbb{U}} = \eta\mathbf{D} - \mathbf{W}$ and the normalized Laplacian $\mathbf{L}_{\mathbb{N}}$ as $\mathbf{L}_{\mathbb{N}} = \eta\mathbf{I}_n - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ with $\eta = 1.01$, as suggested in [3]. Since higher order regularization may be effective on graph-based SSL [11], we used the iterated Laplacian $\mathbf{L}_{\mathbb{I}}$ in our method. Since using $\mathbf{L}_{\mathbb{N}}$ in our method may lead to better results than using $\mathbf{L}_{\mathbb{U}}$ in general SSL tasks [10], we chose $\mathbf{L}_{\mathbb{N}}$ as the "basis" Laplacian. The Laplacian's degree p was chosen in the set $\{1, 2, 3, 4, 5\}$, as suggested in [8].

We set $\mathbf{\Sigma}_{\mathcal{U}} = \mathbf{O}_{u \times u}$ and $\mathbf{\Sigma}_{\mathcal{L}} = \mu\mathbf{I}_l$ for our method. This setting achieved better classification performance on most data sets in comparison to $\mathbf{\Sigma} = \mu\mathbf{I}_n$. The values of μ were chosen in the set $\{0.01, 0.05, 0.1, 0.5, 1, 2, 5, 10, 50, 100\}$, as suggested in [3] for the LGC algorithm.

B. Best case analysis

In this analysis, we evaluate the best error rates of the SSL algorithms over all parameter values on each data set and partition[9]. Table I shows the best average error rates with corresponding standard deviations achieved by our method and the competing SSL algorithms in the partitions of 10 and 100 labeled examples, respectively. In order to statistically evaluate our results, we ran the Friedman test[10] with the Bonferroni post test using our method as "control" algorithm with a significance level of 0.05. We applied the software Orange[11] to run this statistical test using the average rankings shown in Table I. In this table, the rankings for the SSL algorithms that were statistically outperformed by our method on a given partition are marked with a grey background.

Table I shows that our method achieved the best performance in most cases. Moreover, our method showed competitive results in the other three settings. In the G-241C and G-241N data sets, RMGTHOR and CLGC outperformed the competing methods by a large margin in both partitions. These results evidentiate the effectiveness of higher order regularization on SSL.

Our method achieved the best average ranking in both partitions, statistically outperforming GFHF in the partitions of 10 labeled examples. Additional statistical differences might be evidenced if we run experiments on more data sets.

C. Evaluation of classifier stability

In the evaluation of classifier stability [3], we analyze the classification performance of the SSL algorithms with predefined regularization parameters for a given graph construction method with respect to k. For such purpose, we look to the best classification performance and keep the corresponding values of the regularization parameters. Then, we analyze the SSL algorithm's classification performance with respect to k for these predefined regularization parameters.

Fig. 1 shows the average error rates of the SSL algorithms with respect to k in the partitions of 10 labeled examples. In the USPS data set[12], our method achieved competitive results to the other SSL algorithms for most values of k. Moreover, our method showed similar performance to RMGTHOR for $k \leq 10$ and $k \geq 30$.

In the $COIL_2$ data set, our method achieved the best performance for *all* values of k. In the DIGIT-1 data set, our method outperformed the other SSL algorithms for $k \geq 10$. Moreover, our method outperformed GFHF, LGC, LapRLS, and LapSVM by a large margin for $k \geq 10$. Furthermore, the constrained methods (RMGT, RMGTHOR, and CLGC) showed almost stable performances with respect to k.

In the G-241C and G-241N data sets, our method showed similar performance to RMGTHOR, achieving better results than the other SSL algorithms for most values of k. In the TEXT data set, the constrained methods showed similar performance with respect to k, outperforming the other SSL algorithms for $k \geq 10$.

[8] The symKNN-RBF graph is widely used in the SSL literature [2], [1].

[9] We used the same partitions proposed in [1].

[10] See [21] for a review on statistical tests for machine learning.

[11] http://orange.biolab.si/.

[12] USPS is an unbalanced data set with two classes such that the majority class has four times more examples than the minority class. Therefore, we consider that USPS is a data set with *high unbalanced ratio*.

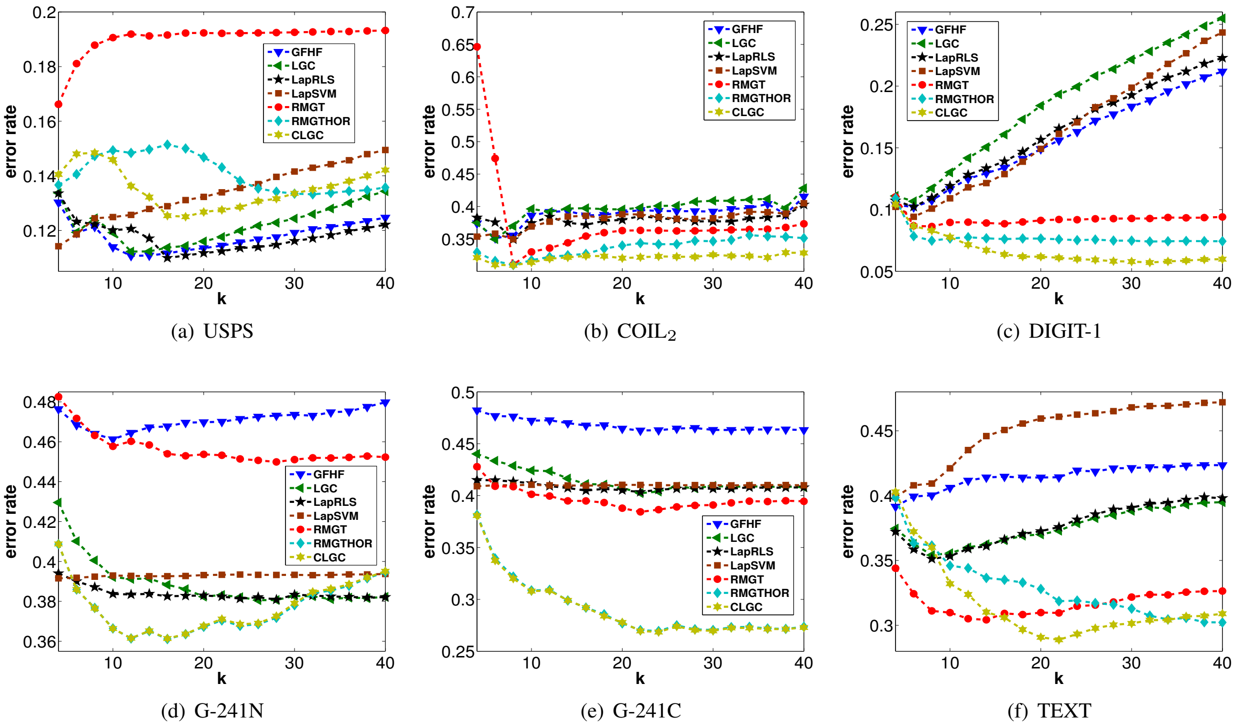

Fig. 1. Average error rates of the SSL algorithms with respect to k using the symKNN-RBF graph in the partitions of 10 labeled examples.

V. Conclusion

We provided a novel graph-based SSL algorithm based on the LGC algorithm, incorporating the normalization constraints in [6] into its regularization framework. We proved that our method admits closed-form solution and generalizes RMGT and RMGTHOR. Through experiments on benchmark data sets, we showed that our method achieves good to exceptional classification performance in comparison to state-of-the-art graph-based SSL algorithms.

Acknowledgments

This research was supported by the Brazilian agencies CAPES and CNPq under grants 303083/2013-1 and 446330/2014-0.

References

[1] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-supervised learning*. The MIT Press, 2006.

[2] X. Zhu, "Semi-supervised learning literature survey," Computer Sciences, University of Wisconsin-Madison, Tech. Rep. 1530, 2005.

[3] C. A. R. Sousa, S. O. Rezende, and G. E. A. P. A. Batista, "Influence of graph construction on semi-supervised learning," in *ECML/PKDD*, 2013, pp. 160–175.

[4] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *NIPS*, 2004, pp. 321–328.

[5] C. A. R. Sousa, V. M. A. Souza, and G. E. A. P. A. Batista, "Time series transductive classification on imbalanced data sets: an experimental study," in *ICPR*, 2014, pp. 3780–3785.

[6] W. Liu and S.-F. Chang, "Robust multi-class transductive learning with graphs," in *CVPR*, 2009, pp. 381–388.

[7] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *ICML*, 2003, pp. 912–919.

[8] C. A. R. Sousa and G. E. A. P. A. Batista, "Robust multi-class graph transduction with higher order regularization," in *IJCNN*, 2015, pp. 299–306.

[9] C. A. R. Sousa, V. M. A. Souza, and G. E. A. P. A. Batista, "An experimental analysis on time series transductive classification on graphs," in *IJCNN*, 2015, pp. 307–314.

[10] R. Johnson and T. Zhang, "On the effectiveness of laplacian normalization for graph semi-supervised learning," *JMLR*, vol. 8, pp. 1489–1517, 2007.

[11] X. Zhou and M. Belkin, "Semi-supervised learning by higher order regularization," in *AISTATS*, 2011, pp. 892–900.

[12] M. Hein, J.-Y. Audibert, and U. von Luxburg, "Graph laplacians and their convergence on random neighborhood graphs," *JMLR*, vol. 8, pp. 1325–1368, 2007.

[13] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[14] C. A. R. Sousa, "An overview on the gaussian fields and harmonic functions method for semi-supervised learning," in *IJCNN*, 2015, pp. 1439–1446.

[15] ——, "An inductive semi-supervised learning approach for the local and global consistency algorithm," in *IJCNN*, 2016.

[16] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: a geometric framework for learning from labeled and unlabeled examples." *JMLR*, vol. 7, pp. 2399–2434, 2006.

[17] K. Zhang, J. T. Kwok, and B. Parvin, "Prototype vector machine for large scale semi-supervised learning," in *ICML*, 2009, pp. 1233–1240.

[18] W. Liu, J. He, and S.-F. Chang, "Large graph construction for scalable semi-supervised learning," in *ICML*, 2010, pp. 679–686.

[19] C. A. R. Sousa, "Impacto da geração de grafos na classificação semissupervisionada," Master's thesis, Universidade de São Paulo, 2013.

[20] T. Jebara, J. Wang, and S.-F. Chang, "Graph construction and b-matching for semi-supervised learning," in *ICML*, 2009, pp. 441–448.

[21] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *JMLR*, vol. 7, pp. 1–30, 2006.