UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

Constrained graph-based semi-supervised learning with higher order regularization

Celso André Rodrigues de Sousa

Tese de Doutorado do Programa de Pós-Graduação em Ciências de
Computação e Matemática Computacional (PPG-CCMC)

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Celso André Rodrigues de Sousa

Constrained graph-based semi-supervised learning with higher order regularization

Doctoral dissertation submitted to the Institute of Mathematics and Computer Sciences – ICMC-USP, in partial fulfillment of the requirements for the degree of the Doctorate Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Gustavo Enrique de Almeida Prado
Alves Batista

USP – São Carlos
October 2017

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

SS725c Sousa, Celso André Rodrigues de
 Constrained graph-based semissupervised learning
 / Celso André Rodrigues de Sousa; orientadora
 Gustavo Enrique de Almeida Prado Alves Batista. --
 São Carlos, 2017.
 158 p.

 Tese (Doutorado - Programa de Pós-Graduação em
 Ciências de Computação e Matemática Computacional) --
 Instituto de Ciências Matemáticas e de Computação,
 Universidade de São Paulo, 2017.

 1. semi-supervised learning. 2. graph-based
 methods. 3. constrained optimization. 4. higher
 order regularization. I. Batista, Gustavo Enrique
 de Almeida Prado Alves, orient. II. Título.

Celso André Rodrigues de Sousa

Aprendizado semissupervisionado restrito baseado em grafos com regularização de ordem elevada

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Gustavo Enrique de Almeida Prado Alves Batista

USP – São Carlos
Outubro de 2017

ABSTRACT

SOUSA, C. A. R. **Constrained graph-based semi-supervised learning with higher order regularization.** 2017. 158 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2017.

Graph-based semi-supervised learning (SSL) algorithms have been widely studied in the last few years. Most of these algorithms were designed from unconstrained optimization problems using a Laplacian regularizer term as smoothness functional in an attempt to reflect the intrinsic geometric structure of the data's marginal distribution. Although a number of recent research papers are still focusing on unconstrained methods for graph-based SSL, a recent statistical analysis showed that many of these algorithms may be unstable on transductive regression. Therefore, we focus on providing new constrained methods for graph-based SSL. We begin by analyzing the regularization framework of existing unconstrained methods. Then, we incorporate two normalization constraints into the optimization problem of three of these methods. We show that the proposed optimization problems have closed-form solution. By generalizing one of these constraints to any distribution, we provide generalized methods for constrained graph-based SSL. The proposed methods have a more flexible regularization framework than the corresponding unconstrained methods. More precisely, our methods can deal with any graph Laplacian and use higher order regularization, which is effective on general SSL tasks. In order to show the effectiveness of the proposed methods, we provide comprehensive experimental analyses. Specifically, our experiments are subdivided into two parts. In the first part, we evaluate existing graph-based SSL algorithms on time series data to find their weaknesses. In the second part, we evaluate the proposed constrained methods against six state-of-the-art graph-based SSL algorithms on benchmark data sets. Since the widely used best case analysis may hide useful information concerning the SSL algorithms' performance with respect to parameter selection, we used recently proposed empirical evaluation models to evaluate our results. Our results show that our methods outperforms the competing methods on most parameter settings and graph construction methods. However, we found a few experimental settings in which our methods showed poor performance. In order to facilitate the reproduction of our results, the source codes, data sets, and experimental results are freely available.

Keywords: semi-supervised learning, graph-based methods, constrained optimization, higher order regularization.

RESUMO

SOUSA, C. A. R. **Aprendizado semissupervisionado restrito baseado em grafos com regularização de ordem elevada**. 2017. 158 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2017.

Algoritmos de aprendizado semissupervisionado baseado em grafos foram amplamente estudados nos últimos anos. A maioria desses algoritmos foi projetada a partir de problemas de otimização sem restrições usando um termo regularizador Laplaciano como funcional de suavidade numa tentativa de refletir a estrutura geométrica intrínseca da distribuição marginal dos dados. Apesar de vários artigos científicos recentes continuarem focando em métodos sem restrição para aprendizado semissupervisionado em grafos, uma análise estatística recente mostrou que muitos desses algoritmos podem ser instáveis em regressão transdutiva. Logo, nós focamos em propor novos métodos com restrições para aprendizado semissupervisionado em grafos. Nós começamos analisando o framework de regularização de métodos sem restrições existentes. Então, nós incorporamos duas restrições de normalização no problema de otimização de três desses métodos. Mostramos que os problemas de otimização propostos possuem solução de forma fechada. Ao generalizar uma dessas restrições para qualquer distribuição, provemos métodos generalizados para aprendizado semissupervisionado restrito baseado em grafos. Os métodos propostos possuem um framework de regularização mais flexível que os métodos sem restrições correspondentes. Mais precisamente, nossos métodos podem lidar com qualquer Laplaciano em grafos e usar regularização de ordem elevada, a qual é efetiva em tarefas de aprendizado semissupervisionado em geral. Para mostrar a efetividade dos métodos propostos, nós provemos análises experimentais robustas. Especificamente, nossos experimentos são subdivididos em duas partes. Na primeira parte, avaliamos algoritmos de aprendizado semissupervisionado em grafos existentes em dados de séries temporais para encontrar possíveis fraquezas desses métodos. Na segunda parte, avaliamos os métodos restritos propostos contra seis algoritmos de aprendizado semissupervisionado baseado em grafos do estado da arte em conjuntos de dados benchmark. Como a amplamente usada análise de melhor caso pode esconder informações relevantes sobre o desempenho dos algoritmos de aprendizado semissupervisionado com respeito à seleção de parâmetros, nós usamos modelos de avaliação empírica recentemente propostos para avaliar os nossos resultados. Nossos resultados mostram que os nossos métodos superam os demais métodos na maioria das configurações de parâmetro e métodos de construção de grafos. Entretanto, encontramos algumas configurações experimentais nas quais nossos métodos mostraram baixo desempenho. Para facilitar a reprodução dos nossos resultados, os códigos fonte, conjuntos de dados e resultados experimentais estão disponíveis gratuitamente.

Palavras-chave: aprendizado semissupervisionado, métodos baseados em grafos, otimização

restrita, regularização de ordem elevada.

LIST OF FIGURES

Figure 1 – Illustrative examples for the following empirical evaluation models: (a) <i>evaluation of classifier stability</i> ; (b) <i>evaluation of graph stability</i> ; and (c) <i>evaluation of error surfaces</i>	74
Figure 2 – Error rates for the SSL algorithms with respect to k on the <i>Gun-Point</i> (using symKNN-HM), <i>50words</i> (using symKNN-LLE), and <i>OSULeaf</i> (using symKNN-HM) data sets.	83
Figure 3 – Error rates for the SSL algorithms with respect to k on the <i>OSULeaf</i> (using symKNN-RBF), <i>Cricket-X</i> (using symFKNN-HM), and <i>Cricket-Y</i> (using symFKNN-RBF) data sets.	84
Figure 4 – Error rates for the SSL algorithms with respect to k on the <i>SonyAIBORobot-Surface</i> (using symFKNN-LLE), <i>CinC-ECG-torso</i> (using symKNN-HM), and <i>ECGFiveDays</i> (using symFKNN-HM) data sets.	85
Figure 5 – Error rates for the SSL algorithms with respect to k on the <i>MedicalImages</i> (using symFKNN-LLE), <i>Wafer</i> (using symKNN-LLE), and <i>StarLightCurves</i> (using symKNN-RBF) data sets.	85
Figure 6 – Error rates for the SSL algorithms with respect to k on the <i>Trace</i> (using symKNN-LLE), <i>FaceFour</i> (using symFKNN-RBF), and <i>Symbols</i> (using symFKNN-LLE) data sets.	86
Figure 7 – Error rates for the SSL algorithms with respect to k on the <i>Adiac</i> , <i>DiatomSizeReduction</i> , and <i>NonInvasiveFetalECG-Thorax2</i> data sets using the mutKNN-RBF graph.	86
Figure 8 – Error rates for the graph construction methods on the <i>SonyAIBORobotSurface</i> (using LapRLS), <i>FacesUCR</i> (using RMGT), and <i>uWaveGestureLibrary-X</i> (using GFHF) data sets.	87
Figure 9 – Error rates for the graph construction methods with respect to k on the <i>50words</i> (using GFHF), <i>MedicalImages</i> (using RMGT), and <i>WordsSynonyms</i> (using GFHF) data sets.	88
Figure 10 – Error rates for the graph construction methods with respect to k on the <i>CinC-ECG-torso</i> (using LGC), <i>ECGFiveDays</i> (using LGC), and <i>FaceFour</i> (using LapRLS) data sets.	88
Figure 11 – Error rates for the graph construction methods with respect to k on the <i>ChlorineConcentration</i> (using GFHF), <i>Gun-Point</i> (using RMGT), and <i>Coffee</i> (using LGC) data sets.	89

Figure 12 – Error surfaces for LGC with respect to k and μ on the <i>MoteStrain</i> (using symKNN-HM), <i>ItalyPowerDemand</i> (using symKNN-RBF), and <i>ChlorineConcentration</i> (using mutKNN-LLE) data sets.	89
Figure 13 – Error surfaces for LGC with respect to k and μ on the <i>FaceFour</i> (using symKNN-LLE), <i>Gun-Point</i> (using mutKNN-LLE), and <i>InlineSkate</i> (using symKNN-LLE) data sets.	90
Figure 14 – Error surfaces for LapRLS with respect to γ_A and γ_I on the <i>SonyAIBORobot-Surface</i> (using symKNN-HM), <i>Gun-Point</i> (using symKNN-RBF), <i>Mallat</i> (using symFKNN-HM) data sets.	90
Figure 15 – Error surfaces for LapRLS with respect to γ_A and γ_I on the <i>Symbols</i> (using mutKNN-HM), <i>CinC-ECG-torso</i> (using mutKNN-RBF), and <i>Wafer</i> (using symFKNN-HM) data sets.	91
Figure 16 – Error surfaces for LapRLS with respect to γ_A and γ_I on the <i>ChlorineConcentration</i> (using mutKNN-RBF), <i>ECGFiveDays</i> (using mutKNN-LLE), and <i>SonyAIBORobotSurface</i> (using mutKNN-HM) data sets.	91
Figure 17 – Error surfaces for LapRLS with respect to γ_A and γ_I on the <i>50words</i> (using symFKNN-HM), <i>Cricket-X</i> (using symKNN-HM), and <i>FaceFour</i> (using symKNN-LLE) data sets.	91
Figure 18 – Error surfaces for LapRLS with respect to γ_A and γ_I on the <i>MoteStrain</i> (using mutKNN-RBF), <i>InlineSkate</i> (using symKNN-LLE), and <i>NonInvasiveFetalECG-ThoraxI</i> (using symFKNN-LLE) data sets.	92
Figure 19 – Error surfaces for LapRLS with respect to k and γ_A on the <i>Adiac</i> (using symFKNN-LLE), <i>ChlorineConcentration</i> (using mutKNN-LLE), and <i>Face-sUCR</i> (using mutKNN-RBF) data sets.	92
Figure 20 – Error surfaces for LapRLS with respect to k and γ_A on the <i>50words</i> (using symKNN-RBF), <i>CinC-ECG-torso</i> (using mutKNN-HM), and <i>Cricket-X</i> (using symKNN-LLE) data sets.	93
Figure 21 – Error surfaces for LapRLS with respect to k and γ_A on the <i>Two-Patterns</i> (using symFKNN-LLE), <i>NonInvasiveFetalECG-ThoraxI</i> (using symFKNN-LLE), and <i>MedicalImages</i> (using mutKNN-RBF) data sets.	93
Figure 22 – Error surfaces for LapRLS with respect to k and γ_I on the <i>SonyAIBORobot-Surface</i> (using mutKNN-RBF), <i>StarLightCurves</i> (using mutKNN-LLE), and <i>SwedishLeaf</i> (using symFKNN-RBF) data sets.	94
Figure 23 – Error surfaces for LapRLS with respect to k and γ_I on the <i>OSULeaf</i> (using symFKNN-HM), <i>MedicalImages</i> (using symFKNN-LLE), and <i>InlineSkate</i> (using symFKNN-LLE) data sets.	94
Figure 24 – Error surfaces for LapRLS with respect to k and γ_I on the <i>Cricket-X</i> (using symFKNN-LLE), <i>MoteStrain</i> (using symKNN-LLE), and <i>TwoLeadECG</i> (using mutKNN-RBF) data sets.	95

Figure 25 – Average error rates for the SSL algorithms with respect to k using symKNN-RBF in the partitions of 10 labeled examples.	108
Figure 26 – Average error rates for the SSL algorithms with respect to k on G-241C and G-241N using mutKNN-based graphs in the partitions of 10 labeled examples.	109
Figure 27 – Average error rates for the SSL algorithms with respect to k on G-241C and G-241N using mutKNN-based graphs in the partitions of 100 labeled examples.	110
Figure 28 – Error surfaces for CGFHF with respect to k and p using symKNN-RBF in the partitions of 10 labeled examples.	112
Figure 29 – Error surfaces for CLGC with respect to k and p using symKNN-RBF in the partitions of 10 labeled examples.	113
Figure 30 – Error surfaces for CLGC with respect to k and μ using symKNN-RBF in the partitions of 10 labeled examples.	114
Figure 31 – Error surfaces for CLGC with respect to k and μ on G-241C using RBF-based graphs in the partitions of 10 labeled examples.	114
Figure 32 – Error surfaces for CLGC with respect to p and μ using symKNN-RBF in the partitions of 10 labeled examples.	115
Figure 33 – Error surfaces for CLapRLS with respect to k and p using symKNN-RBF in the partitions of 10 labeled examples.	116
Figure 34 – Error surfaces for CLapRLS with respect to γ_A and γ_I using symKNN-RBF in the partitions of 10 labeled examples.	116
Figure 35 – Error surfaces for CLapRLS with respect to γ_A and γ_I on G-241C using LLE-based graphs in the partitions of 10 labeled examples.	117
Figure 36 – Error surfaces for CLapRLS with respect to k and γ_A using symKNN-RBF in the partitions of 10 labeled examples.. . . .	118
Figure 37 – Error surfaces for CLapRLS with respect to k and γ_I using symKNN-RBF in the partitions of 10 labeled examples.	118
Figure 38 – Error surfaces for CLapRLS with respect to p and γ_A using symKNN-RBF in the partitions of 10 labeled examples.	119
Figure 39 – Error surfaces for CLapRLS with respect to p and γ_I using symKNN-RBF in the partitions of 10 labeled examples.	120

LIST OF ALGORITHMS

Algoritmo 1 – CGFHF for transduction	48
Algoritmo 2 – Computing $\mathbf{X} = \mathbf{DC}$	52
Algoritmo 3 – CLGC for transduction	54
Algoritmo 4 – CLapRLS for induction	59
Algoritmo 5 – MINRES algorithm for K-GFHF	143

LIST OF TABLES

Table 1	– Description of the time series data sets used in the first experiment.	68
Table 2	– Description of the data sets used in the second experiment.	69
Table 3	– Error rates (%) for the SSL algorithms for each graph construction method on the D1–D15 data sets.	78
Table 4	– Error rates (%) for the SSL algorithms for each graph construction method on the D16–D30 data sets.	79
Table 5	– Error rates (%) for the SSL algorithms for each graph construction method on the D31–D45 data sets.	80
Table 6	– Average rankings for the SSL algorithms for each graph construction method.	82
Table 7	– Average rankings for the graph construction methods for each SSL algorithm.	82
Table 8	– Best error rates (%) for each SSL algorithm using the results shown in Fig. 3(a).	84
Table 9	– Best average error rates (%) and standard deviations (%) for the SSL algorithms using symKNN-RBF.	104
Table 10	– Best average error rates (%) and standard deviations (%) for the SSL algorithms using mutKNN-RBF.	104
Table 11	– Best average error rates (%) and standard deviations (%) for the SSL algorithms using symFKNN-RBF.	105
Table 12	– Best average error rates (%) and standard deviations (%) for the SSL algorithms using symKNN-HM.	105
Table 13	– Best average error rates (%) and standard deviations (%) for the SSL algorithms using mutKNN-HM.	106
Table 14	– Best average error rates (%) and standard deviations (%) for the SSL algorithms using symFKNN-HM.	106
Table 15	– Best average error rates (%) and standard deviations (%) for the SSL algorithms using symKNN-LLE.	107
Table 16	– Best average error rates (%) and standard deviations (%) for the SSL algorithms using mutKNN-LLE.	107
Table 17	– Best average error rates (%) and standard deviations (%) for the SSL algorithms using symFKNN-LLE.	108
Table 18	– Best average error rates (%) for the SSL algorithms in the experimental settings described in Fig. 26 and 27.	111

CONTENTS

1	INTRODUCTION	21
1.1	Problems in graph-based semi-supervised learning	22
1.1.1	<i>Naive experimental evaluations</i>	22
1.1.2	<i>Focus on unconstrained optimization</i>	23
1.2	Main hypothesis	24
1.3	Main contributions	24
1.4	Outline of the thesis	25
2	GRAPH-BASED SEMI-SUPERVISED LEARNING	27
2.1	Notation	27
2.2	Gaussian Fields and Harmonic Functions (GFHF)	29
2.3	Local and Global Consistency (LGC)	30
2.4	Laplacian Regularized Least Squares (LapRLS)	32
2.5	Laplacian Support Vector Machine (LapSVM)	32
2.6	Robust Multi-class Graph Transduction (RMGT)	33
2.7	Concluding remarks	34
3	THEORETICAL ANALYSIS ON GRAPH-BASED SEMI-SUPERVISED LEARNING	35
3.1	Analysis of GFHF	35
3.2	Analysis of LGC	37
3.3	Inductive forms of GFHF and LGC	40
3.4	Our findings	40
4	CONSTRAINED METHODS FOR GRAPH-BASED SEMI-SUPERVISED LEARNING	43
4.1	Constrained Gaussian Fields and Harmonic Functions (CGFHF)	44
4.1.1	<i>Regularization framework</i>	44
4.1.2	<i>Properties</i>	46
4.1.3	<i>Computing CGFHF</i>	47
4.2	Constrained Local and Global Consistency (CLGC)	48
4.2.1	<i>Regularization framework</i>	49
4.2.2	<i>Properties</i>	50
4.2.3	<i>Computing CLGC</i>	52

4.3	Constrained Laplacian Regularized Least Squares (CLapRLS)	54
4.3.1	<i>Regularization framework</i>	54
4.3.2	<i>Properties</i>	56
4.3.3	<i>Computing CLapRLS</i>	58
4.4	Generalization to any distribution	59
4.4.1	<i>Generalized CGFHF</i>	60
4.4.2	<i>Generalized CLGC</i>	62
4.4.3	<i>Generalized CLapRLS</i>	63
4.5	Our findings	65
5	EXPERIMENTAL SETUP	67
5.1	Sets of experiments	67
5.2	Graph construction	69
5.2.1	<i>Distance function</i>	69
5.2.2	<i>Adjacency graph construction</i>	69
5.2.3	<i>Weighted matrix generation</i>	70
5.3	Graph Laplacian generation	72
5.4	Graph-based SSL algorithms	72
5.5	Empirical evaluation models	73
5.5.1	<i>Best case analysis</i>	73
5.5.2	<i>Evaluation of classifier stability</i>	74
5.5.3	<i>Evaluation of graph stability</i>	75
5.5.4	<i>Evaluation of error surfaces</i>	75
5.6	Concluding remarks	76
6	TIME SERIES TRANSDUCTIVE CLASSIFICATION ON GRAPHS	77
6.1	Best case analysis	77
6.2	Evaluation of classifier stability	83
6.3	Evaluation of graph stability	87
6.4	Evaluation of error surfaces	89
6.4.1	<i>Error surfaces for LGC ($k \times \mu$)</i>	89
6.4.2	<i>Error surfaces for LapRLS ($\gamma_A \times \gamma_I$)</i>	90
6.4.3	<i>Error surfaces for LapRLS ($k \times \gamma_A$)</i>	92
6.4.4	<i>Error surfaces for LapRLS ($k \times \gamma_I$)</i>	94
6.5	Discussion	95
6.5.1	<i>Effectiveness of mutKNN-based graphs</i>	95
6.5.2	<i>Effectiveness of LLE-based graphs</i>	96
6.5.3	<i>Uneffectiveness of LGC</i>	97
6.5.4	<i>Adaptive kernel size for RMGT</i>	97

6.5.5	<i>Uneffectiveness of RMGT on data sets with high unbalanced ratio</i>	98
6.5.6	<i>Numerical instabilities of RMGT</i>	98
6.5.7	<i>Effectiveness of LapRLS</i>	98
6.5.8	<i>Parameter selection for LapRLS</i>	99
6.5.9	<i>Kernel selection for LapRLS</i>	99
6.6	<i>Our findings</i>	100
7	EXPERIMENTS ON CONSTRAINED GRAPH-BASED SEMI-SUPERVISED LEARNING	103
7.1	Best case analysis	103
7.2	Evaluation of classifier stability	108
7.3	Evaluation of error surfaces	111
7.3.1	<i>Error surfaces for CGFHF ($k \times p$)</i>	111
7.3.2	<i>Error surfaces for CLGC ($k \times p$)</i>	112
7.3.3	<i>Error surfaces for CLGC ($k \times \mu$)</i>	113
7.3.4	<i>Error surfaces for CLGC ($p \times \mu$)</i>	114
7.3.5	<i>Error surfaces for CLapRLS ($k \times p$)</i>	115
7.3.6	<i>Error surfaces for CLapRLS ($\gamma_A \times \gamma_I$)</i>	115
7.3.7	<i>Error surfaces for CLapRLS ($k \times \gamma_A$)</i>	117
7.3.8	<i>Error surfaces for CLapRLS ($k \times \gamma_I$)</i>	117
7.3.9	<i>Error surfaces for CLapRLS ($p \times \gamma_A$)</i>	119
7.3.10	<i>Error surfaces for CLapRLS ($p \times \gamma_I$)</i>	120
7.4	<i>Our findings</i>	120
8	CONCLUSION	123
	BIBLIOGRAPHY	125
APPENDIX A	HEURISTIC INDUCTIVE SOLUTION FOR GFHF	133
APPENDIX B	KERNEL APPROACH FOR GFHF	137
B.1	Iterative solution for K-GFHF	139
B.2	Speeding up K-GFHF using Krylov subspaces	141
APPENDIX C	KERNEL APPROACH FOR CGFHF	145
APPENDIX D	HEURISTIC INDUCTIVE SOLUTION FOR LGC	149
D.1	Reformulation of LGC	149
D.2	iLGC	155

INTRODUCTION

In many machine learning applications, one frequently encounters situations in which only a very restricted number of labeled examples is available while large amounts of unlabeled examples can be easily obtained at increasing rates. Acquiring labeled examples may be costly, time-consuming, tedious, and usually requires expert knowledge. In addition, labeled examples are usually much harder (or expensive) to obtain than unlabeled examples.

For instance, it is unfeasible to accurately annotate sufficient number of web pages of all categories of interest to obtain large enough fully labeled data sets for web page classification, necessary for supervised learning tasks. Therefore, classifiers that can effectively deal with unlabeled examples are required on a variety of real applications. However, this may not be an easy task, even when we consider only the task of *transductive classification*¹.

We consider in this thesis a *static* scenario. Specifically, we obtain sets of labeled and unlabeled examples and our focus is to classify the unlabeled examples in the training sample without the necessity to update our model when novel (labeled or unlabeled) examples arrive.

Assuming that both labeled and unlabeled examples are generated by some fixed (though unknown) underlying distribution, unlabeled examples usually can reveal useful information related to that data distribution (URNER; SHALEV-SHWARTZ; BEN-DAVID, 2011). At least intuitively, we can use such an information to generate classifiers that can effectively learn from both labeled and unlabeled examples.

This intuition has driven the field of *semi-supervised learning*² (SSL), which has gained increasing attention in the last few years (ZHU, 2005; CHAPELLE; SCHÖLKOPF; ZIEN,

¹ Transductive classification refers to machine learning tasks in which we generate a classification function from both labeled and unlabeled examples using (scarce) label information without providing generalization for the entire sample space. Therefore, transductive classification is a kind of semi-supervised classification.

² In this thesis, we consider *semi-supervised learning* a synonym of *semi-supervised classification*. This is highly accepted in the SSL literature (ZHU, 2005; CHAPELLE; SCHÖLKOPF; ZIEN, 2006).

2006; LIU *et al.*, 2009; LIU; HE; CHANG, 2010; BELKIN; NIYOGI; SINDHWANI, 2006; ZHOU *et al.*, 2004; ZHU; GHAMRANI; LAFFERTY, 2003). A variety of effective SSL algorithms have been proposed in the literature (JOACHIMS, 2003; ZHOU; LI, 2005; SZUMMER; JAAKKOLA, 2001; SZUMMER; JAAKKOLA, 2002; SINDHWANI; KEERTHI, 2006; ROSSET *et al.*, 2005; MANN; MCCALLUM, 2007; LAWRENCE; JORDAN, 2005; KRISHNAPURAM *et al.*, 2005; GRANDVALET; BENGIO, 2005; CHAPELLE; ZIEN, 2005), each one of them with their own assumptions on the data. Among all SSL algorithms, graph-based methods are widely used due to their promising *performances*³ on a number of application domains (ZHU; GHAMRANI; LAFFERTY, 2003; CHAPELLE; SCHÖLKOPF; ZIEN, 2006; SOUSA; REZENDE; BATISTA, 2013; SOUSA; SOUZA; BATISTA, 2014; SOUSA; SOUZA; BATISTA, 2015).

A variety of graph-based SSL algorithms solve a (usually) *convex* optimization problem (also known as *regularization framework*), possibly subject to some fitting and/or normalization constraints, using a Laplacian regularizer term (in quadratic form) as smoothness functional. Formally, this term is a smoothness penalty term that tries to reflect the intrinsic geometric structure of the data's marginal distribution from a weighted graph that encodes similarities between training examples.

1.1 Problems in graph-based semi-supervised learning

We now describe the two problems in graph-based SSL we faced in this thesis.

1.1.1 Naive experimental evaluations

The research community has proposed a number of graph-based SSL algorithms (TONG; JIN, 2007; BELKIN; NIYOGI; SINDHWANI, 2006; LIU *et al.*, 2009; ZHOU *et al.*, 2004; ZHU; GHAMRANI; LAFFERTY, 2003; WU *et al.*, 2012; ZHANG; HUANG; LIU, 2011; ZHANG; ZHOU, 2009; ZHU; LAFFERTY, 2005) as well as a number of graph construction methods (HEIN; MAIER, 2007; LIU *et al.*, 2009; ROWEIS; SAUL, 2000; DAITCH; KELNER; SPIELMAN, 2009; NIE *et al.*, 2011; ZHUANG *et al.*, 2012; GHANVININEJAD *et al.*, 2011; LUO *et al.*, 2012; WANG; XIA, 2012; ZASS; SHASHUA, 2006; HUANG; JEBARA, 2007; HUANG; JEBARA, 2011). Despite the recent theoretical advances on graph-based SSL, the *empirical maturity* of this research field has not grown enough in order to provide insights to practitioners on the SSL algorithms' performance for real applications.

Many research papers provide empirical comparisons of graph-based SSL algorithms based only on their best error rates (or other measure) on each data set over all parameter values and graph construction methods (when applicable). This procedure is called *best case analysis*

³ In this thesis, we consider that the term “performance” is always related to *classification performance*. More precisely, we consider that performance is equivalent to *error rate* in our experiments.

(SOUSA; REZENDE; BATISTA, 2013). Several empirical comparisons of graph-based SSL algorithms using only the best case analysis on a *single* (or a few) weighted graph(s) over all parameter values (when applicable) have been done in the SSL literature.

Although this is the most commonly used experimental analysis used in the SSL literature (ZHU, 2005; CHAPELLE; SCHÖLKOPF; ZIEN, 2006), it may not provide all the necessary information to choose the best classifiers for real applications. In addition, it may hide useful information concerning the classifiers' *stability*, leading to (possibly) tendentious conclusions related to the classifiers' performance. Therefore, the best case analysis is not adequate for a comprehensive experimental analysis. We confirm this statement in our experiments on benchmark data sets.

We consider that an SSL algorithm is stable for a given graph construction method if there is no abrupt changes in its classification performance with respect to parameter selection. For instance, (almost) stable classifiers may be preferable over classifiers which are able to provide excellent performance only for a very narrow range of parameter values and mediocre performance for the remaining values. Therefore, the *empirical stability* of the classifiers for a given graph construction method with respect to parameter selection may be crucial for the success of graph-based SSL on real applications.

The problem of evaluating the empirical stability of graph-based SSL algorithms was firstly addressed by Sousa, Rezende and Batista (2013). The authors proposed a robust experimental setup and three evaluation models that are capable to evaluate the algorithms over a number of graph construction methods and parameter settings. In this thesis, we use the experimental setup in (SOUSA; REZENDE; BATISTA, 2013) in our experiments to provide comprehensive experimental evaluations on graph-based SSL. Moreover, we extend their evaluation models and provide insights on how they can be effectively used on future research.

1.1.2 Focus on unconstrained optimization

Most graph-based SSL algorithms were designed from unconstrained optimization problems (ZHU; GHAMRANI; LAFFERTY, 2003; ZHOU *et al.*, 2004; BELKIN; NIYOGI; SINDHWANI, 2006). These methods have a good mathematical background with interesting theoretical properties.

Despite the popularity of unconstrained methods for graph-based SSL, the statistical analysis in (CORTES *et al.*, 2008) showed that some of these methods are unstable for the task of *transductive regression*⁴. This can hinder the effective use of these methods on real applications.

⁴ Transductive regression is equivalent to transductive classification, but the labels are real numbers instead of discrete ones. Several other algorithms introduced for transductive classification can be viewed in fact as transductive regression ones as their objective function is based on the squared loss, e.g. (ZHU; GHAMRANI; LAFFERTY, 2003; ZHOU *et al.*, 2004; BELKIN; NIYOGI; SINDHWANI, 2006).

[Cortes et al. \(2008\)](#) used the notion of algorithmic stability to derive novel algorithm-dependent generalization bounds for several families of transductive regression algorithms, both by using convexity and closed-form solutions. These bounds generalize the stability bounds given in ([BOUSQUET; ELISSEEFF, 2002](#)) for the inductive setting and extend to regression the stability-based transductive classification bounds in ([EL-YANIV; PECHYONY, 2006](#)).

Specifically, [Cortes et al. \(2008\)](#) analyzed the stability of the transductive regression algorithms in ([ZHU; GHAHRAMANI; LAFFERTY, 2003](#); [ZHOU et al., 2004](#); [BELKIN; NIYOGI; SINDHWANI, 2006](#); [WU; Schölkopf, 2007](#)) based on their closed form solution and propose a modification to the seemingly unstable algorithm that makes them stable and guarantees a non-trivial generalization bound. Moreover, the authors also proved that the original algorithms can be stable if we incorporate a constraint on the function in the optimization problem.

To summarize, [Cortes et al. \(2008\)](#) proved that unconstrained methods for graph-based SSL may be unstable. Moreover, they proved that these methods can be stable if we incorporate a common constraint into the optimization problem. Therefore, the use of constrained optimization on graph-based SSL can be really effective.

Although a number of recent research papers are still focusing on unconstrained methods for graph-based SSL ([YANG; COHEN; SALAKHUDINOV, 2016](#); [SUBRAMANYA; BILMES, 2011](#); [LIU; YANG, 2015](#); [FANG; CHANG; LAUW, 2014](#); [SOLOMON et al., 2014](#); [SHIVANNA et al., 2015](#)), we focus on constrained optimization in this thesis. Specifically, we propose three new constrained methods for graph-based SSL that lead to state-of-the-art performance on a number of benchmark data sets. However, we found no generalization bound for our methods.

1.2 Main hypothesis

Hypothesis 1 states the main hypothesis of this thesis.

Hypothesis 1. *The addition of the normalization constraints of [Liu and Chang \(2009\)](#) into the optimization problem of unconstrained graph-based SSL algorithms leads to better performance in comparison to the original unconstrained methods for most experimental settings.*

1.3 Main contributions

The main contributions of this thesis are summarized as follows:

- we propose three new graph-based SSL algorithms applying the normalization constraints of [Liu and Chang \(2009\)](#) into the optimization problem of existing methods;
- we prove that the proposed constrained methods are naturally inductive;

- we propose optimized ways to compute our methods under some conditions;
- we generalize one of the constraints of [Liu and Chang \(2009\)](#) to *any* distribution. From this new constraint, we generalized the proposed constrained methods to *any* distribution;
- we show the effectiveness of our methods on benchmark data sets over many graph construction methods and parameter settings.

Additional contributions are described in Chapters [3](#), [4](#), [6](#), and [7](#) and in the appendix.

1.4 Outline of the thesis

This thesis is organized as follows:

- Chapter [2](#) describes the notation used throughout the thesis and provides a background on the graph-based SSL algorithms we used in our experiments;
- Chapter [3](#) provides a theoretical analysis on the algorithms in ([ZHU; GHAMRAMANI; LAFFERTY, 2003](#); [ZHOU et al., 2004](#)), proving that they are inductive methods, as opposed to their original formulations;
- Chapter [4](#) provides three new constrained methods based on the algorithms in ([ZHU; GHAMRAMANI; LAFFERTY, 2003](#); [ZHOU et al., 2004](#); [BELKIN; NIYOGI; SINDHWANI, 2006](#)) by incorporating the normalization constraints of [Liu and Chang \(2009\)](#) into their optimization problem. Moreover, we generalize one of these constraints to *any* distribution. From this new constraint, we provide generalized methods for constrained graph-based SSL;
- Chapter [5](#) describes the experimental setup we used in our experiments;
- Chapter [6](#) provides our experimental analysis on time series transductive classification on graphs over 45 data sets from UCR repository⁵. The supplementary material containing all the results for this experiment is freely available⁶;
- Chapter [7](#) evaluates the proposed constrained methods against state-of-the-art graph-based SSL algorithms on the benchmark data sets in ([CHAPELLE; SCHÖLKOPF; ZIEN, 2006](#))⁷. The supplementary material containing all the results for this experiment is freely available⁸;

⁵ http://www.cs.ucr.edu/~eamonn/time_series_data/.

⁶ https://www.researchgate.net/publication/316649218_Time_series_transductive_classification_on_graphs_supplementary_material.

⁷ <http://olivier.chapelle.cc/ssl-book/benchmarks.html>.

⁸ https://www.researchgate.net/publication/316634288_Constrained_graph-based_semi-supervised_learning_supplementary_material.

- Chapter 8 summarizes the main findings in this thesis.

GRAPH-BASED SEMI-SUPERVISED LEARNING

This chapter revises five state-of-the-art graph-based SSL algorithms. We revise the regularization framework of these algorithms as well as their iterative solutions, when applicable.

Our review is based on the review in (SOUSA; REZENDE; BATISTA, 2013) with additional comments. Specifically, we revise in this chapter the following algorithms:

- *Gaussian Fields and Harmonic Functions* (GFHF) (ZHU; GHAHRAMANI; LAFFERTY, 2003);
- *Local and Global Consistency* (LGC) (ZHOU *et al.*, 2004);
- *Laplacian Regularized Least Squares* (LapRLS) (BELKIN; NIYOGI; SINDHWANI, 2006);
- *Laplacian Support Vector Machine* (LapSVM) (BELKIN; NIYOGI; SINDHWANI, 2006);
- *Robust Multi-class Graph Transduction* (RMGT) (LIU; CHANG, 2009).

This chapter is organized as follows. Section 2.1 describes the notation for graph-based SSL. Sections 2.2, 2.3, 2.4, 2.5, 2.6 revise GFHF, LGC, LapRLS, LapSVM, and RMGT, respectively. Finally, Section 2.7 describes some remarks on the SSL algorithms we revised in this chapter.

2.1 Notation

Consider a training sample $\mathcal{X} := \{\mathbf{x}_i\}_{i=1}^n \subset \mathbb{R}^d$ in which just the first l examples are labeled, i.e., $\mathbf{x}_i$ has label $y_i \in \mathbb{N}_c$ where $\mathbb{N}_a := \{i \in \mathbb{N}^* | 1 \leq i \leq a\}$ with $a \in \mathbb{N}^*$ and $c \in \mathbb{N}^*$ being

the number of classes. Let $u := n - l$ be the amount of unlabeled examples and $\mathbf{Y} \in \mathbb{B}^{n \times c}$ be a label matrix in which $\mathbf{Y}_{ij} = 1$ if and only if $\mathbf{x}_i$ has label $y_i = j$. Consider $\mathcal{X}_{\mathcal{L}} := \{\mathbf{x}_i\}_{i=1}^l$ and $\mathcal{X}_{\mathcal{U}} := \{\mathbf{x}_i\}_{i=l+1}^n$ the sets of labeled and unlabeled examples, respectively.

Consider an undirected graph $\mathcal{G} := (\mathcal{X}, \mathcal{E})$ in which each $\mathbf{x}_i$ is a node of $\mathcal{G}$. Let $\mathcal{N}_i \subset \mathcal{X}$ be the set of neighbors of $\mathbf{x}_i$ and $\mathbf{x}_i^{(k)} \in \mathcal{X}$ the k -th nearest neighbor of $\mathbf{x}_i$. In order to generate a sparse weighted matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$ from $\mathcal{G}$ one uses a similarity function $\mathcal{K} : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$ to compute the weights between pairs of neighbors $(\mathbf{x}_i, \mathbf{x}_j)$.

Let $\mathbf{F} \in \mathbb{R}^{n \times c}$ be the output of a given graph-based SSL algorithm. All matrices can be subdivided into labeled and unlabeled submatrices. The $\mathbf{F}$ and $\mathbf{Y}$ matrices are subdivided into two submatrices while all others are subdivided into four submatrices. For instance:

$$\mathbf{W} := \begin{bmatrix} \mathbf{W}_{\mathcal{L}\mathcal{L}} & \mathbf{W}_{\mathcal{L}\mathcal{U}} \\ \mathbf{W}_{\mathcal{U}\mathcal{L}} & \mathbf{W}_{\mathcal{U}\mathcal{U}} \end{bmatrix} \quad \mathbf{Y} := \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ \mathbf{Y}_{\mathcal{U}} \end{bmatrix}$$

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

Given a weighted matrix $\mathbf{W}$, we have to generate a graph Laplacian¹ in order to start the regularization process or perform classification. The *combinatorial* Laplacian is defined by:

$$\mathbf{L}_{\mathbb{C}} := \mathbf{D} - \mathbf{W} \quad (2.1)$$

such that $\mathbf{D} := \text{diag}(\mathbf{W}\mathbf{1}_n)$ is the diagonal degree matrix. By definition, we have $\mathbf{D}_{ii} = \sum_{j=1}^n \mathbf{W}_{ij}$, $\forall i \in \mathbb{N}_n$. The *normalized* Laplacian is defined by:

$$\mathbf{L}_{\mathbb{N}} := \mathbf{D}^{-1/2} \mathbf{L}_{\mathbb{C}} \mathbf{D}^{-1/2} = \mathbf{I}_n - \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2} \quad (2.2)$$

such that $\mathbf{I}_n$ is the n -by- n identity matrix. Let $\mathbf{P} := \mathbf{D}^{-1} \mathbf{W} \in \mathbb{R}^{n \times n}$ be the probability transition matrix.

Assume that the superscript $^\top$ denotes transposition of matrices or vectors. Given a vector $\mathbf{f} := [f(\mathbf{x}_1), f(\mathbf{x}_2), \dots, f(\mathbf{x}_n)]^\top \in \mathbb{R}^n$, in which $f : \mathbb{R}^d \mapsto \mathbb{R}$, we can write:

$$\begin{aligned} \mathbf{f}^\top \mathbf{L}_{\mathbb{C}} \mathbf{f} &= \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \mathbf{W}_{ij} (f(\mathbf{x}_i) - f(\mathbf{x}_j))^2 \\ \mathbf{f}^\top \mathbf{L}_{\mathbb{N}} \mathbf{f} &= \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \mathbf{W}_{ij} \left(\frac{f(\mathbf{x}_i)}{\sqrt{\mathbf{D}_{ii}}} - \frac{f(\mathbf{x}_j)}{\sqrt{\mathbf{D}_{jj}}} \right)^2 \end{aligned}$$

¹ See (HEIN; AUDIBERT; LUXBURG, 2007) for a general formulation of graph Laplacians. For simplicity, we only used here the matrix formulations of these operators.

Then, $\mathbf{L}_{\mathbb{C}}$ and $\mathbf{L}_{\mathbb{N}}$ are *positive semidefinite* (p.s.d.) matrices ($\mathbf{L}_{\mathbb{C}} \succcurlyeq 0$ and $\mathbf{L}_{\mathbb{N}} \succcurlyeq 0$), i.e., symmetric matrices with nonnegative eigenvalues. Therefore, the objective functions of most graph-based SSL algorithms become convex in $\mathbf{f}$. This means we can obtain their closed-form solutions in most cases.

2.2 Gaussian Fields and Harmonic Functions (GFHF)

GFHF (ZHU; GHAHRAMANI; LAFFERTY, 2003) can be viewed as the following optimization problem:

$$\min_{\mathbf{F} \in \mathbb{R}^{u \times c}} \text{tr}(\mathbf{F}^\top \mathbf{L} \mathbf{F}) \quad \text{s.t.} \quad \mathbf{F}_{\mathcal{L}} = \mathbf{Y}_{\mathcal{L}} \quad (2.3)$$

where $\text{tr}(\mathbf{X}) := \sum_{i=1}^a \mathbf{X}_{ii}$, $\forall a \in \mathbb{N}^*$, is the trace of a matrix $\mathbf{X} \in \mathbb{R}^{a \times a}$. If $\mathbf{F}_{\mathcal{L}} = \mathbf{Y}_{\mathcal{L}}$, the objective function of (2.3) can be rewritten as:

$$\text{tr}(\mathbf{F}^\top \mathbf{L} \mathbf{F}) = \text{tr}(\mathbf{Y}_{\mathcal{L}}^\top \mathbf{L}_{\mathcal{L}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}) + \text{tr}(\mathbf{F}_{\mathcal{U}}^\top \mathbf{L}_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}}) + 2\text{tr}(\mathbf{F}_{\mathcal{U}}^\top \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}) \quad (2.4)$$

Therefore, minimization problem (2.3) can be rewritten as:

$$\min_{\mathbf{F}_{\mathcal{U}} \in \mathbb{R}^{u \times c}} \text{tr}(\mathbf{F}_{\mathcal{U}}^\top \mathbf{L}_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}}) + 2\text{tr}(\mathbf{F}_{\mathcal{U}}^\top \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}) \quad (2.5)$$

The closed-form solution of (2.3) (using the solution of (2.5)) is given by:

$$\mathbf{F} = \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \end{bmatrix} \quad (2.6)$$

Zhu and Ghahramani (2002) showed that, for $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$, we can solve (2.5) iteratively, independent of the initial matrix $\mathbf{F}_{\mathcal{U}}^{(0)} \in \mathbb{R}^{u \times c}$. We describe this algorithm in Proposition 2.1.

Proposition 2.1 (*Iterative solution of GFHF for $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$*). *Given an initial matrix $\mathbf{F}_{\mathcal{U}}^{(0)} \in \mathbb{R}^{u \times c}$, minimization problem (2.5) can be solved for $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$ using the following recursion:*

$$\mathbf{F}_{\mathcal{U}}^{(t+1)} = \mathbf{P}_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}}^{(t)} + \mathbf{P}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \quad (2.7)$$

Proof. Iterating (2.7), we have:

$$\mathbf{F}_{\mathcal{U}}^{(t)} = (\mathbf{P}_{\mathcal{U}\mathcal{U}})^t \mathbf{F}_{\mathcal{U}}^{(0)} + \left(\sum_{i=0}^{t-1} (\mathbf{P}_{\mathcal{U}\mathcal{U}})^i \right) \mathbf{P}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}$$

For $t \rightarrow +\infty$, we have:

$$\begin{aligned} \lim_{t \rightarrow +\infty} (\mathbf{P}_{\mathcal{U}\mathcal{U}})^t &= \mathbf{O}_{u \times u} \\ \lim_{t \rightarrow +\infty} \sum_{i=0}^{t-1} (\mathbf{P}_{\mathcal{U}\mathcal{U}})^i &= (\mathbf{I}_u - \mathbf{P}_{\mathcal{U}\mathcal{U}})^{-1} \end{aligned} \quad (2.8)$$

Then,

$$\mathbf{F}_{\mathcal{U}}^* = \lim_{t \rightarrow +\infty} \mathbf{F}_{\mathcal{U}}^{(t)} = (\mathbf{I}_u - \mathbf{P}_{\mathcal{U}\mathcal{U}})^{-1} \mathbf{P}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \quad (2.9)$$

Since $\mathbf{P} = \mathbf{D}^{-1} \mathbf{W}$, we can write:

$$\begin{aligned} \begin{bmatrix} \mathbf{D}_{\mathcal{L}} & \mathbf{O}_{l \times u} \\ \mathbf{O}_{u \times l} & \mathbf{D}_{\mathcal{U}} \end{bmatrix} \begin{bmatrix} \mathbf{P}_{\mathcal{L}\mathcal{L}} & \mathbf{P}_{\mathcal{L}\mathcal{U}} \\ \mathbf{P}_{\mathcal{U}\mathcal{L}} & \mathbf{P}_{\mathcal{U}\mathcal{U}} \end{bmatrix} &= \begin{bmatrix} \mathbf{W}_{\mathcal{L}\mathcal{L}} & \mathbf{W}_{\mathcal{L}\mathcal{U}} \\ \mathbf{W}_{\mathcal{U}\mathcal{L}} & \mathbf{W}_{\mathcal{U}\mathcal{U}} \end{bmatrix} \\ \begin{bmatrix} \mathbf{D}_{\mathcal{L}} \mathbf{P}_{\mathcal{L}\mathcal{L}} & \mathbf{D}_{\mathcal{L}} \mathbf{P}_{\mathcal{L}\mathcal{U}} \\ \mathbf{D}_{\mathcal{U}} \mathbf{P}_{\mathcal{U}\mathcal{L}} & \mathbf{D}_{\mathcal{U}} \mathbf{P}_{\mathcal{U}\mathcal{U}} \end{bmatrix} &= \begin{bmatrix} \mathbf{W}_{\mathcal{L}\mathcal{L}} & \mathbf{W}_{\mathcal{L}\mathcal{U}} \\ \mathbf{W}_{\mathcal{U}\mathcal{L}} & \mathbf{W}_{\mathcal{U}\mathcal{U}} \end{bmatrix} \end{aligned} \quad (2.10)$$

This yields $\mathbf{P}_{\mathcal{U}\mathcal{L}} = \mathbf{D}_{\mathcal{U}}^{-1} \mathbf{W}_{\mathcal{U}\mathcal{L}}$ and $\mathbf{P}_{\mathcal{U}\mathcal{U}} = \mathbf{D}_{\mathcal{U}}^{-1} \mathbf{W}_{\mathcal{U}\mathcal{U}}$. Therefore, (2.9) can be rewritten as:

$$\begin{aligned} \mathbf{F}_{\mathcal{U}}^* &= (\mathbf{I}_u - \mathbf{P}_{\mathcal{U}\mathcal{U}})^{-1} \mathbf{P}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} = (\mathbf{I}_u - \mathbf{D}_{\mathcal{U}}^{-1} \mathbf{W}_{\mathcal{U}\mathcal{U}})^{-1} \mathbf{D}_{\mathcal{U}}^{-1} \mathbf{W}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ &= (\mathbf{D}_{\mathcal{U}} - \mathbf{W}_{\mathcal{U}\mathcal{U}})^{-1} \mathbf{W}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \end{aligned}$$

Since $(\mathbf{L}_{\mathcal{C}})_{\mathcal{U}\mathcal{U}} = \mathbf{D}_{\mathcal{U}} - \mathbf{W}_{\mathcal{U}\mathcal{U}}$ and $(\mathbf{L}_{\mathcal{C}})_{\mathcal{U}\mathcal{L}} = -\mathbf{W}_{\mathcal{U}\mathcal{L}}$, we have:

$$\mathbf{F}_{\mathcal{U}}^* = -(\mathbf{L}_{\mathcal{C}})_{\mathcal{U}\mathcal{U}}^{-1} (\mathbf{L}_{\mathcal{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}$$

□

2.3 Local and Global Consistency (LGC)

LGC (ZHOU *et al.*, 2004) can be viewed as the following optimization problem:

$$\min_{\mathbf{F} \in \mathbb{R}^{n \times c}} \text{tr} \left(\mathbf{F}^{\top} \mathbf{L} \mathbf{F} + (\mathbf{F} - \mathbf{Y})^{\top} \Sigma (\mathbf{F} - \mathbf{Y}) \right) \quad (2.11)$$

where $\Sigma := \begin{bmatrix} \Sigma_{\mathcal{L}} & \mathbf{O}_{l \times u} \\ \mathbf{O}_{u \times l} & \Sigma_{\mathcal{U}} \end{bmatrix} \in \mathbb{R}^{n \times n}$ is a diagonal matrix that contains the regularization parameters. $\Sigma_{\mathcal{L}} \in \mathbb{R}^{l \times l}$ and $\Sigma_{\mathcal{U}} \in \mathbb{R}^{u \times u}$ are diagonal matrices that encode the regularization parameters

for the labeled and unlabeled examples, respectively. In the original formulation, we have $\Sigma = \mu \mathbf{I}_n$ where $\mu \in \mathbb{R}_+^*$ is the “global” regularization parameter. The closed-form solution of (2.11) is given by:

$$\mathbf{F} = (\mathbf{L} + \Sigma)^{-1} \Sigma \mathbf{Y} \quad (2.12)$$

Zhou *et al.* (2004) showed that, for $\Sigma = \mu \mathbf{I}_n$ and $\mathbf{L} = \mathbf{L}_{\mathbb{N}}$, we can solve (2.11) iteratively, independent of the initial matrix $\mathbf{F}^{(0)} \in \mathbb{R}^{n \times c}$. We describe this algorithm in Proposition 2.2.

Proposition 2.2 (*Iterative solution of LGC for $\Sigma = \mu \mathbf{I}_n$ and $\mathbf{L} = \mathbf{L}_{\mathbb{N}}$*). *Given an initial matrix $\mathbf{F} \in \mathbb{R}^{n \times c}$, minimization problem (2.11) can be solved for $\Sigma = \mu \mathbf{I}_n$ and $\mathbf{L} = \mathbf{L}_{\mathbb{N}}$ using the following recursion:*

$$\mathbf{F}^{(t+1)} = \frac{1}{1+\mu} \mathbf{S} \mathbf{F}^{(t)} + \frac{\mu}{1+\mu} \mathbf{Y} \quad (2.13)$$

in which $\mathbf{S} = \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2}$.

Proof. Iterating (2.13), we have:

$$\mathbf{F}^{(t)} = \left(\frac{1}{1+\mu} \mathbf{S} \right)^t \mathbf{F}^{(0)} + \frac{\mu}{1+\mu} \left(\sum_{i=0}^{t-1} \left(\frac{1}{1+\mu} \mathbf{S} \right)^i \right) \mathbf{Y} \quad (2.14)$$

Since the eigenvalues of $\mathbf{S}$ are in the interval $[-1, 1]$ (CHUNG, 1997), we have, for $t \rightarrow +\infty$:

$$\begin{aligned} \lim_{t \rightarrow +\infty} \left(\frac{1}{1+\mu} \mathbf{S} \right)^t &= \mathbf{O}_{n \times n} \\ \lim_{t \rightarrow +\infty} \left(\sum_{i=0}^{t-1} \left(\frac{1}{1+\mu} \mathbf{S} \right)^i \right) &= \left(\mathbf{I}_n - \frac{1}{1+\mu} \mathbf{S} \right)^{-1} \mathbf{Y} \end{aligned}$$

Therefore,

$$\mathbf{F}^* = \lim_{t \rightarrow +\infty} \mathbf{F}^{(t)} = \frac{\mu}{1+\mu} \left(\mathbf{I}_n - \frac{1}{1+\mu} \mathbf{S} \right)^{-1} \mathbf{Y}$$

By definition, we have:

$$\mathbf{L}_{\mathbb{N}} = \mathbf{I}_n - \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2} = \mathbf{I}_n - \mathbf{S}$$

This yields:

$$\mathbf{F}^* = \lim_{t \rightarrow +\infty} \mathbf{F}^{(t)} = \left(\mathbf{I}_n + \frac{1}{\mu} \mathbf{L}_{\mathbb{N}} \right)^{-1}$$

□

2.4 Laplacian Regularized Least Squares (LapRLS)

The multi-class version of LapRLS (BELKIN; NIYOGI; SINDHWANI, 2006) can be viewed as the following optimization problem:

$$\min_{\alpha \in \mathbb{R}^{n \times c}} \text{tr} \left((\mathbf{K}\alpha - \mathbf{Y})^\top \Sigma (\mathbf{K}\alpha - \mathbf{Y}) + \gamma_A \alpha^\top \mathbf{K} \alpha + \gamma_l \alpha^\top \mathbf{K} \mathbf{L} \mathbf{K} \alpha \right) \quad (2.15)$$

where $\mathbf{K} \in \mathbb{R}^{n \times n}$ is a kernel matrix and γ_A and γ_l are the regularization parameters. The closed-form solution of (2.15) is given by:

$$\alpha = (\Sigma \mathbf{K} + \gamma_A \mathbf{I}_n + \gamma_l \mathbf{L} \mathbf{K})^{-1} \Sigma \mathbf{Y} \quad (2.16)$$

Then, we get the output matrix $\mathbf{F} = \mathbf{K}\alpha$ for transduction. Through kernel expansions over both labeled and unlabeled examples, we can compute the output vector $\mathbf{f}^x \in \mathbb{R}^c$ for any $\mathbf{x} \in \mathbb{R}^d$ as:

$$\mathbf{f}_j^x = \sum_{i=1}^n \mathcal{K}(\mathbf{x}, \mathbf{x}_i) \alpha_{ij}, \quad \forall j \in \mathbb{N}_c \quad (2.17)$$

2.5 Laplacian Support Vector Machine (LapSVM)

LapSVM (BELKIN; NIYOGI; SINDHWANI, 2006) can be viewed as the following optimization problem:

$$\min_{f \in \mathcal{H}_{\mathcal{K}}} \frac{1}{l} \sum_{i=1}^l h(y_i f(\mathbf{x}_i)) + \gamma_A \|f\|_{\mathcal{H}_{\mathcal{K}}} + \gamma_l \mathbf{f}^\top \mathbf{L} \mathbf{f} \quad (2.18)$$

where $h(x) = \max(0, 1 - x)$ is called *hinge loss*, $\mathcal{H}_{\mathcal{K}}$ is the *Reproducing Kernel Hilbert Space (RKHS)* for the kernel function $\mathcal{K}$, $\|\cdot\|_{\mathcal{H}_{\mathcal{K}}}$ is the norm in $\mathcal{H}_{\mathcal{K}}$, and $\mathbf{f} := [f(\mathbf{x}_1), \dots, f(\mathbf{x}_n)]^\top \in \mathbb{R}^n$. Let $\mathbf{y} := [y_1, \dots, y_l, 0, \dots, 0] \in \mathbb{R}^n$ be a label vector in which $y_i \in \{-1, +1\}$, $\forall i \in \mathbb{N}_l$. As in LapRLS, γ_A and γ_l are the regularization parameters.

Through kernel expansions over both labeled and unlabeled examples, the solution of (2.18) for any $\mathbf{x} \in \mathbb{R}^d$ can be written as:

$$f(\mathbf{x}) = \sum_{i=1}^n \mathcal{K}(\mathbf{x}, \mathbf{x}_i) \alpha_i$$

with $\alpha \in \mathbb{R}^n$. Therefore, the closed-form solution of (2.18) is given by:

$$\alpha = \frac{1}{2} (\gamma_A \mathbf{I}_n + \gamma_L \mathbf{L} \mathbf{K})^{-1} \bar{\mathbf{J}}^\top \bar{\mathbf{Y}} \beta^* \quad (2.19)$$

where $\bar{\mathbf{Y}} := \text{diag}([y_1, \dots, y_l]^\top) \in \mathbb{R}^{l \times l}$ is a label matrix and $\bar{\mathbf{J}} := [\mathbf{I}_l \quad \mathbf{0}_{l \times u}] \in \mathbb{R}^{l \times n}$ is a “selection” matrix. The vector $\beta^* \in \mathbb{R}^l$ is given by:

$$\begin{aligned} \beta^* &= \arg \min_{\beta \in \mathbb{R}^l} \mathbf{1}_l^\top \beta - \frac{1}{2} \beta^\top \mathbf{Q} \beta \\ \text{s.t. } &\mathbf{y}^\top \beta = 0, \quad 0 \leq \beta \leq \frac{1}{l} \end{aligned}$$

such that $\mathbf{Q} \in \mathbb{R}^{l \times l}$ is given by:

$$\mathbf{Q} = \frac{1}{2} \bar{\mathbf{Y}} \bar{\mathbf{J}} \mathbf{K} (\gamma_A \mathbf{I}_n + \gamma_L \mathbf{L} \mathbf{K})^{-1} \bar{\mathbf{J}}^\top \bar{\mathbf{Y}}$$

2.6 Robust Multi-class Graph Transduction (RMGT)

Liu and Chang (2009) proposed the following two normalization constraints for graph-based SSL: (1) $\mathbf{F} \mathbf{1}_c = \mathbf{1}_n$; and (2) $\mathbf{F}^\top \mathbf{1}_n = n\omega$ where $\omega \in \mathbb{R}^c$ is a model’s parameter². The first constraint imposes that $\mathbf{F}$ is row-normalized, which is equivalent to $\sum_{j=1}^c \mathbf{F}_{ij} = 1, \forall i \in \mathbb{N}_n$. Since the authors did not impose that $\mathbf{F} \geq 0$, $\mathbf{F}_{ik}$ acts like a posterior pseudo-probability $p(\mathbb{C}_k | \mathbf{x}_i)$ where $\mathbb{C}_k$ represents the k -th class. Knowing the class priors $\omega := [p(\mathbb{C}_1), p(\mathbb{C}_2), \dots, p(\mathbb{C}_c)]^\top$ and assuming that the marginal probability density $p(\mathbf{x}_i)$ is approximated by a uniform distribution ($p(\mathbf{x}_i) \approx 1/n$), we would have the following equation for the k -th class (LIU; CHANG, 2009):

$$p(\mathbb{C}_k) = \sum_{i=1}^n p(\mathbf{x}_i) p(\mathbb{C}_k | \mathbf{x}_i) \approx \frac{1}{n} \sum_{i=1}^n p(\mathbb{C}_k | \mathbf{x}_i) = \frac{1}{n} \sum_{i=1}^n \mathbf{F}_{ik} = \frac{1}{n} \mathbf{1}_n^\top \mathbf{F}_{:,k} \quad (2.20)$$

This leads to the constraint $\mathbf{1}_n^\top \mathbf{F} = n\omega^\top$, which is equivalent to $\mathbf{F}^\top \mathbf{1}_n = n\omega$. This constraint enforces output normalization using class prior knowledge assuming that $p(\mathbf{x}_i)$ is generated by a uniform distribution.

² ω can be the class prior probabilities or the uniform class distribution ($\omega = \mathbf{1}_c/c$), as suggested by (LIU; CHANG, 2009). ω is *not* a parameter for tuning.

RMGT (LIU; CHANG, 2009) is equivalent to GFHF (see (2.3)) for $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$ subject to $\mathbf{F}\mathbf{1}_c = \mathbf{1}_n$ and $\mathbf{F}^\top \mathbf{1}_n = n\omega$. Mathematically, RMGT can be viewed as the following constrained optimization problem:

$$\begin{aligned} \min_{\mathbf{F} \in \mathbb{R}^{n \times c}} \quad & \text{tr}(\mathbf{F}^\top \mathbf{L}_{\mathbb{C}} \mathbf{F}) \\ \text{s.t.} \quad & \mathbf{F}_{\mathcal{L}} = \mathbf{Y}_{\mathcal{L}}, \quad \mathbf{F}\mathbf{1}_c = \mathbf{1}_n, \quad \mathbf{F}^\top \mathbf{1}_n = n\omega \end{aligned} \quad (2.21)$$

If $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$ in (2.4), we have:

$$\text{tr}(\mathbf{F}^\top \mathbf{L}_{\mathbb{C}} \mathbf{F}) = \text{tr}(\mathbf{Y}_{\mathcal{L}}^\top (\mathbf{L}_{\mathbb{C}})_{\mathcal{L}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}) + \text{tr}(\mathbf{F}_{\mathcal{U}}^\top (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}}) + 2\text{tr}(\mathbf{F}_{\mathcal{U}}^\top (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}})$$

Therefore, minimization problem (2.21) can be rewritten as:

$$\begin{aligned} \min_{\mathbf{F}_{\mathcal{U}} \in \mathbb{R}^{u \times c}} \quad & \text{tr}(\mathbf{F}_{\mathcal{U}}^\top (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}}) + 2\text{tr}(\mathbf{F}_{\mathcal{U}}^\top (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}) \\ \text{s.t.} \quad & \mathbf{F}_{\mathcal{U}}\mathbf{1}_c = \mathbf{1}_u, \quad \mathbf{F}_{\mathcal{U}}^\top \mathbf{1}_u = n\omega - \mathbf{Y}_{\mathcal{L}}^\top \mathbf{1}_l \end{aligned} \quad (2.22)$$

As shown in (LIU; CHANG, 2009), the closed-form solution of (2.21) (from the solution of (2.22)) is given by:

$$\mathbf{F} = \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \boldsymbol{\zeta} \end{bmatrix} \quad (2.23)$$

in which

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

2.7 Concluding remarks

We revised in this chapter five state-of-the-art graph-based SSL algorithms. In the next chapter, we will provide our theoretical analysis for these methods with interesting theoretical properties. We focus our attention on scalable approaches and strategies for out-of-sample extension.

THEORETICAL ANALYSIS ON GRAPH-BASED SEMI-SUPERVISED LEARNING

In this chapter, we describe our theoretical analysis on graph-based SSL. Specifically, we analyze GFHF and LGC, focusing on their regularization frameworks, convergence properties, and approaches to speed up these methods.

This chapter is organized as follows. Section 3.1 shows our analysis for GFHF. Section 3.2 shows our analysis for LGC. Section 3.3 shows that GFHF and LGC are special cases of LapRLS. Finally, Section 3.4 summarizes our theoretical findings.

3.1 Analysis of GFHF

Given an initial matrix $\mathbf{F}_{\mathcal{U}}^{(0)} \in \mathbb{R}^{u \times c}$, the method in (2.7) iterates until the convergence is reached, considering that $\mathbf{F}_{\mathcal{L}}^{(t)} = \mathbf{Y}_{\mathcal{L}}, \forall t \in \mathbb{N}^*$. In Proposition 3.1, we prove that (2.7) always converges when $\sum_{j=1}^l (\mathbf{P}_{\mathcal{U}\mathcal{L}})_{i-l,j} > 0, \forall \mathbf{x}_i \in \mathcal{X}_{\mathcal{U}}$. This restriction is equivalent to $\mathbf{P}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l > 0$.

Proposition 3.1 (Convergence restriction for GFHF). *If $\mathbf{P}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l > 0$, (2.7) converges to $\mathbf{F}_{\mathcal{U}}^* = (\mathbf{I}_u - \mathbf{P}_{\mathcal{U}\mathcal{U}})^{-1} \mathbf{P}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}$, independently of $\mathbf{F}_{\mathcal{U}}^{(0)} \in \mathbb{R}^{u \times c}$.*

Proof. Iterating (2.7) for an initial matrix $\mathbf{F}_{\mathcal{U}}^{(0)} \in \mathbb{R}^{u \times c}$, we have, $\forall t \in \mathbb{N}^*$:

$$\mathbf{F}_{\mathcal{U}}^{(t)} = (\mathbf{P}_{\mathcal{U}\mathcal{U}})^t \mathbf{F}_{\mathcal{U}}^{(0)} + \left(\sum_{i=0}^{t-1} (\mathbf{P}_{\mathcal{U}\mathcal{U}})^i \right) \mathbf{P}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}$$

By definition, $\mathbf{P}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l + \mathbf{P}_{\mathcal{U}\mathcal{U}} \mathbf{1}_u = \mathbf{1}_u$. If $\sum_{j=1}^l (\mathbf{P}_{\mathcal{U}\mathcal{L}})_{i-l,j} > 0$, then $\exists \gamma, 0 < \gamma < 1$, such that $\sum_{j=1}^u (\mathbf{P}_{\mathcal{U}\mathcal{U}})_{i-l,j} \leq \gamma, \forall \mathbf{x}_i \in \mathcal{X}_{\mathcal{U}}$. Therefore, we have $\sum_{j=1}^u [(\mathbf{P}_{\mathcal{U}\mathcal{U}})_{i-l,j}]^t \leq \gamma^t, \forall t \in \mathbb{N}^*$, which yields:

$$\begin{aligned}\lim_{t \rightarrow +\infty} (\mathbf{P}_{\mathcal{U}\mathcal{U}})^t &= \mathbf{O}_{u \times u} \\ \lim_{t \rightarrow +\infty} \sum_{i=0}^{t-1} (\mathbf{P}_{\mathcal{U}\mathcal{U}})^i &= (\mathbf{I}_u - \mathbf{P}_{\mathcal{U}\mathcal{U}})^{-1}\end{aligned}$$

Then,

$$\mathbf{F}_{\mathcal{U}}^* = \lim_{t \rightarrow +\infty} \mathbf{F}_{\mathcal{U}}^{(t)} = (\mathbf{I}_u - \mathbf{P}_{\mathcal{U}\mathcal{U}})^{-1} \mathbf{P}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}$$

□

Proposition 3.1 is based on the proof of Zhu (2005), not restricted to fully connected graphs. This yields that Zhu's proof can not be extended to *sparse* graphs. Therefore, we can not use (2.7) to prove the convergence of GFHF to any weighted graph. Clearly, the graph must have at least one labeled example on each component (SOUSA; REZENDE; BATISTA, 2013).

Proposition 3.2 shows an iterative algorithm to solve (2.3) for *any* graph Laplacian. Such an algorithm can be viewed as the generalized iterative solution for GFHF.

Proposition 3.2 (*Convergence of GFHF for any graph Laplacian*). *Given an initial matrix $\mathbf{F}_{\mathcal{U}}^{(0)} \in \mathbb{R}^{u \times c}$, if the eigenvalues of $\mathbf{I}_u - \mathbf{L}_{\mathcal{U}\mathcal{U}}$ are in the interval $[-1, 1]$, we can solve (2.3) using the following recursion:*

$$\mathbf{F}_{\mathcal{U}}^{(t+1)} = (\mathbf{I}_u - \mathbf{L}_{\mathcal{U}\mathcal{U}}) \mathbf{F}_{\mathcal{U}}^{(t)} - \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \quad (3.1)$$

Proof. Iterating (3.1), we have:

$$\mathbf{F}_{\mathcal{U}}^{(t)} = (\mathbf{I}_u - \mathbf{L}_{\mathcal{U}\mathcal{U}})^t \mathbf{F}_{\mathcal{U}}^{(0)} - \left[\sum_{i=0}^{t-1} (\mathbf{I}_u - \mathbf{L}_{\mathcal{U}\mathcal{U}})^i \right] \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}$$

If the eigenvalues of $\mathbf{I}_u - \mathbf{L}_{\mathcal{U}\mathcal{U}}$ are in the interval $[-1, 1]$, we have:

$$\begin{aligned}\lim_{t \rightarrow +\infty} (\mathbf{I}_u - \mathbf{L}_{\mathcal{U}\mathcal{U}})^t &= \mathbf{O}_{u \times u} \\ \lim_{t \rightarrow +\infty} \sum_{i=0}^{t-1} (\mathbf{I}_u - \mathbf{L}_{\mathcal{U}\mathcal{U}})^i &= \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1}\end{aligned}$$

Therefore,

$$\lim_{t \rightarrow +\infty} \mathbf{F}_{\mathcal{U}}^{(t)} = -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}$$

□

3.2 Analysis of LGC

Proposition 3.3 shows that GFHF is a special case of LGC.

Proposition 3.3 (GFHF as special case of LGC). *If $\Sigma_{\mathcal{U}} = \mathbf{O}_{u \times u}$ and $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$, (2.12) is reduced to (2.6), approximately.*

Proof. Let $\Theta = (\mathbf{L} + \Sigma)^{-1}$. Then, (2.12) can be rewritten as:

$$\mathbf{F} = \begin{bmatrix} \Theta_{\mathcal{L}\mathcal{L}} & \Theta_{\mathcal{L}\mathcal{U}} \\ \Theta_{\mathcal{U}\mathcal{L}} & \Theta_{\mathcal{U}\mathcal{U}} \end{bmatrix} \begin{bmatrix} \Sigma_{\mathcal{L}} & \mathbf{O}_{l \times u} \\ \mathbf{O}_{u \times l} & \mathbf{O}_{u \times u} \end{bmatrix} \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ \mathbf{O}_{u \times c} \end{bmatrix} = \begin{bmatrix} \Theta_{\mathcal{L}\mathcal{L}} \Sigma_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ \Theta_{\mathcal{U}\mathcal{L}} \Sigma_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \end{bmatrix} \quad (3.2)$$

Since $(\mathbf{L} + \Sigma)\Theta = \mathbf{I}_n$, we can write:

$$\begin{bmatrix} \mathbf{L}_{\mathcal{L}\mathcal{L}} + \Sigma_{\mathcal{L}} & \mathbf{L}_{\mathcal{L}\mathcal{U}} \\ \mathbf{L}_{\mathcal{U}\mathcal{L}} & \mathbf{L}_{\mathcal{U}\mathcal{U}} \end{bmatrix} \begin{bmatrix} \Theta_{\mathcal{L}\mathcal{L}} & \Theta_{\mathcal{L}\mathcal{U}} \\ \Theta_{\mathcal{U}\mathcal{L}} & \Theta_{\mathcal{U}\mathcal{U}} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_l & \mathbf{O}_{l \times u} \\ \mathbf{O}_{u \times l} & \mathbf{I}_u \end{bmatrix}$$

This yields:

$$\begin{aligned} \Theta_{\mathcal{L}\mathcal{L}} &= \Gamma \Sigma_{\mathcal{L}}^{-1} \\ \Theta_{\mathcal{U}\mathcal{L}} &= -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \Gamma \Sigma_{\mathcal{L}}^{-1} \\ \Theta_{\mathcal{L}\mathcal{U}} &= -\Gamma \Sigma_{\mathcal{L}}^{-1} \mathbf{L}_{\mathcal{L}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \\ \Theta_{\mathcal{U}\mathcal{U}} &= \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \Gamma \Sigma_{\mathcal{L}}^{-1} \mathbf{L}_{\mathcal{L}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \end{aligned}$$

in which

$$\Gamma = (\Sigma_{\mathcal{L}}^{-1} \mathbf{L}_{\mathcal{L}\mathcal{L}} + \mathbf{I}_l - \Sigma_{\mathcal{L}}^{-1} \mathbf{L}_{\mathcal{L}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}})^{-1}$$

If $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$, $\Gamma \rightarrow \mathbf{I}_l$. Therefore, when $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$, (3.2) yields:

$$\begin{aligned} \mathbf{F} &= \begin{bmatrix} \Theta_{\mathcal{L}\mathcal{L}} \Sigma_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ \Theta_{\mathcal{U}\mathcal{L}} \Sigma_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \end{bmatrix} = \begin{bmatrix} \Gamma \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \Gamma \mathbf{Y}_{\mathcal{L}} \end{bmatrix} \\ &\approx \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \end{bmatrix} \end{aligned}$$

□

Proposition 3.4 provides a generalized method to solve (2.11) with constraints in the matrix $\Sigma^{-1} \mathbf{L}$. This method can be posed as the iterative solution of LGC for any graph Laplacian.

Proposition 3.4 (General iterative solution for LGC). *Given an initial matrix $\mathbf{F}^{(0)} \in \mathbb{R}^{n \times c}$, if the eigenvalues of $\Sigma^{-1}\mathbf{L}$ are in the interval $[-1, 1]$, we can solve (2.11) using the following recursion:*

$$\mathbf{F}^{(t+1)} = -\Sigma^{-1}\mathbf{L}\mathbf{F}^{(t)} + \mathbf{Y} \quad (3.3)$$

Proof. Iterating (3.3), we have:

$$\mathbf{F}^{(t)} = (-\Sigma^{-1}\mathbf{L})^t \mathbf{F}^{(0)} + \left[\sum_{i=0}^{t-1} (-\Sigma^{-1}\mathbf{L})^i \right] \mathbf{Y}$$

If the eigenvalues of $\Sigma^{-1}\mathbf{L}$ are in the interval $[-1, 1]$, we have:

$$\begin{aligned} \lim_{t \rightarrow +\infty} (-\Sigma^{-1}\mathbf{L})^t &= \mathbf{O}_{n \times n} \\ \lim_{t \rightarrow +\infty} \sum_{i=0}^{t-1} (-\Sigma^{-1}\mathbf{L})^i &= (\mathbf{I}_n + \Sigma^{-1}\mathbf{L})^{-1} \end{aligned}$$

Therefore,

$$\lim_{t \rightarrow +\infty} \mathbf{F}^{(t)} = (\mathbf{I}_n + \Sigma^{-1}\mathbf{L})^{-1} \mathbf{Y} = (\mathbf{L} + \Sigma)^{-1} \Sigma \mathbf{Y}$$

□

Proposition 3.5 shows that we can solve (2.11) under some constraints on $\mathbf{L}$ and Σ using two iterative methods, obtaining the same closed-form solution, independently of $\mathbf{F}^{(0)}$. Specifically, Proposition 3.5 shows that (3.3) is reduced to the iterative form of LGC (the iterative solution in (ZHOU *et al.*, 2004)) under some constraints.

Proposition 3.5 (Equivalence to LGC). *Given an initial matrix $\mathbf{F}^{(0)} \in \mathbb{R}^{n \times c}$, if $\mathbf{L} = \mathbf{L}_{\mathbb{N}}$ and $\Sigma = \mu \mathbf{I}_n$ with $\mu \in \mathbb{R}_+^*$, we can solve (2.11) using any of the following recursions:*

$$\mathbf{F}^{(t+1)} = -\frac{1}{\mu} \mathbf{L}_{\mathbb{N}} \mathbf{F}^{(t)} + \mathbf{Y} \quad (3.4)$$

$$\mathbf{F}^{(t+1)} = \frac{1}{1+\mu} \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2} \mathbf{F}^{(t)} + \frac{\mu}{1+\mu} \mathbf{Y} \quad (3.5)$$

Proof. (3.4) comes directly from Proposition 3.4 for $\mathbf{L} = \mathbf{L}_{\mathbb{N}}$ and $\Sigma = \mu \mathbf{I}_n$. (3.5) comes directly from Proposition 2.2.

□

Eq. (3.5) is the iterative form of LGC. Therefore, such a method is a special case of (3.3) for $\mathbf{L} = \mathbf{L}_{\mathbb{N}}$ and $\Sigma = \mu \mathbf{I}_n$. If $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$ and $\Sigma_{\mathcal{U}} \rightarrow \mathbf{O}_{u \times u}$ in (3.3), we can conclude that (3.3) produces the same output of GFHF, as shown in Corollary 3.6.

Corollary 3.6 (Equivalence to GFHF). *If $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$ and $\Sigma_{\mathcal{U}} \rightarrow \mathbf{O}_{u \times u}$ in (3.3), we obtain:*

$$\lim_{t \rightarrow +\infty} \mathbf{F}^{(t)} \approx \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \end{bmatrix}$$

Proof. From (3.3), we can write:

$$\begin{aligned} \mathbf{F}_{\mathcal{L}}^{(t+1)} &= -\Sigma_{\mathcal{L}}^{-1} \mathbf{L}_{\mathcal{L}\mathcal{L}} \mathbf{F}_{\mathcal{L}}^{(t)} - \Sigma_{\mathcal{L}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}}^{\top} \mathbf{F}_{\mathcal{U}}^{(t)} + \mathbf{Y}_{\mathcal{L}} \\ \mathbf{F}_{\mathcal{U}}^{(t+1)} &= -\Sigma_{\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{F}_{\mathcal{L}}^{(t)} - \Sigma_{\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}}^{(t)} \end{aligned} \quad (3.6)$$

Since $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$, we have $\mathbf{F}_{\mathcal{L}}^{(t+1)} \approx \mathbf{Y}_{\mathcal{L}}, \forall t \geq 0$, and also $\lim_{t \rightarrow +\infty} \mathbf{F}_{\mathcal{L}}^{(t)} \approx \mathbf{Y}_{\mathcal{L}}$. Therefore, we have, $\forall t \geq 0$:

$$\mathbf{F}_{\mathcal{U}}^{(t+1)} \approx -\Sigma_{\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{F}_{\mathcal{L}}^{(t)} - \Sigma_{\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{U}} \mathbf{Y}_{\mathcal{L}}$$

This yields:

$$\begin{aligned} \lim_{t \rightarrow +\infty} \mathbf{F}_{\mathcal{U}}^{(t)} &\approx -(\mathbf{I}_u + \Sigma_{\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{U}})^{-1} \Sigma_{\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ &\approx -(\Sigma_{\mathcal{U}} + \mathbf{L}_{\mathcal{U}\mathcal{U}})^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \approx -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \end{aligned}$$

□

In Corollary 3.7, we show that (3.3) is equivalent to the iterative solution in (ZHU; GHAHRAMANI, 2002) under some constraints. Such a method outputs the same solution of GFHF for $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$.

Corollary 3.7 (Equivalence to Zhu's method). *Let $\mathbf{P} := \mathbf{D}^{-1} \mathbf{W} \in \mathbb{R}^{n \times n}$ be the probability transition matrix. If $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$, $\Sigma_{\mathcal{U}} \rightarrow \mathbf{O}_{u \times u}$, and $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$, we can compute (3.3) setting $\mathbf{F}_{\mathcal{L}}^{(t)} = \mathbf{Y}_{\mathcal{L}}, \forall t \in \mathbb{N}^*$, and use any of the following recursions:*

$$\mathbf{F}_{\mathcal{U}}^{(t+1)} = \mathbf{P}_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}}^{(t)} + \mathbf{P}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \quad (3.7)$$

$$\mathbf{F}_{\mathcal{U}}^{(t+1)} = -\Sigma_{\mathcal{U}}^{-1} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}}^{(t)} - \Sigma_{\mathcal{U}}^{-1} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \quad (3.8)$$

Proof. The proof comes directly from Corollary 3.6 and Proposition 2.1.

□

3.3 Inductive forms of GFHF and LGC

This section shows that GFHF and LGC are naturally inductive methods, instead of transductive ones. Specifically, we show that LapRLS generalizes GFHF and LGC. This theoretical result is shown in Proposition 3.8. For the proof, consider the optimization problem of LapRLS in (2.15) and its closed-form solution in (2.16).

Proposition 3.8 (*Inductive forms of GFHF and LGC*). *If $\gamma_A = 0$ and $\gamma_I = 1$, (2.16) is reduced to:*

$$\alpha = \mathbf{K}^{-1}(\mathbf{L} + \Sigma)^{-1}\Sigma\mathbf{Y} \quad (3.9)$$

Therefore, the transductive solution $\mathbf{F} = \mathbf{K}\alpha$ is equivalent to (2.12). Moreover, if $\Sigma_{\mathcal{U}} = \mathbf{O}_{u \times u}$ and $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$ in (3.9), we obtain:

$$\alpha \approx \mathbf{K}^{-1} \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1}\mathbf{L}_{\mathcal{U}\mathcal{L}}\mathbf{Y}_{\mathcal{L}} \end{bmatrix} \quad (3.10)$$

Therefore, $\mathbf{F} = \mathbf{K}\alpha$ is equivalent to (2.6), approximately.

Proof. We obtain (3.9) directly from (2.16). Applying Proposition 3.3 in (3.9), we obtain (3.10). $\square$

Proposition 3.8 shows that (3.9) is the inductive form of LGC while (3.10) is the inductive form of GFHF. Moreover, this proposition shows that the kernel matrix $\mathbf{K}$ has no influence in the transductive solution of these methods.

3.4 Our findings

In this chapter, we provided theoretical analyses on graph-based SSL algorithms. Our findings can be summarized as follows:

1. we showed that the convergence proof in (ZHU; GHAHRAMANI, 2002; ZHU, 2005) can not be extended to sparse graphs. Specifically, we showed in Proposition 3.1 that there is a convergence restriction for GFHF when applying the iterative method in (2.7). When this restriction is satisfied, the convergence of (2.7) is guaranteed using the proof in (ZHU; GHAHRAMANI, 2002; ZHU, 2005);
2. we provided in Proposition 3.2 an iterative method for GFHF for *any* graph Laplacian under certain constraints on $\mathbf{L}$. This method can be viewed as the *generalized iterative method* for GFHF;

3. we proved that GFHF is a special case of LGC. Specifically, we proved that the solution of LGC can be reduced to the solution of GFHF under some conditions;
4. we provided in Proposition 3.4 an iterative method for LGC for *any* graph Laplacian under certain constraints on $\mathbf{L}$ and Σ . This method can be viewed as the *generalized iterative method* for LGC. We also proved that the methods in (ZHU; GHARAMANI, 2002; ZHU, 2005) are special cases of this generalized method;
5. we provided the inductive forms of GFHF and LGC. Mathematically, we showed that LapRLS generalizes GFHF and LGC. This means that these two methods can be expressed by kernel expansions over both labeled and unlabeled examples. Therefore, GFHF and LGC are naturally *inductive*¹ methods, instead of transductive ones.

¹ The original formulations of GFHF and LGC state that these methods are transductive (ZHU; GHARAMANI; LAFFERTY, 2003; ZHOU *et al.*, 2004).

CONSTRAINED METHODS FOR GRAPH-BASED SEMI-SUPERVISED LEARNING

This chapter describes the three constrained methods we created for graph-based SSL and their corresponding generalized forms. Our methods were designed based on GFHF, LGC, and LapRLS combined with the normalization constraints created by [Liu and Chang \(2009\)](#). As opposed to the transductive method of [Liu and Chang \(2009\)](#), we show that our methods are naturally inductive through kernel expansions over both labeled and unlabeled examples. Specifically, we provide in this chapter the following constrained graph-based SSL algorithms:

- *Constrained Gaussian Fields and Harmonic Functions* (CGFHF);
- *Constrained Local and Global Consistency* (CLGC);
- *Constrained Laplacian Regularized Least Squares* (CLapRLS).

CGFHF and CLGC were originally proposed in ([SOUSA; BATISTA, 2015](#); [SOUSA; BATISTA, 2016](#)). The original name of CGFHF was *Robust Multi-class Graph Transduction with Higher Order Regularization* (RMGTHOR). This name was changed because we proved recently that this algorithm is inductive, instead of transductive. Most of the formulations remain the same as in ([SOUSA; BATISTA, 2015](#)).

This chapter is organized as follows. Sections [4.1](#), [4.2](#), [4.3](#) formulate CGFHF, CLGC, and CLapRLS, respectively. Section [4.4](#) generalize CGFHF, CLGC, and CLapRLS to *any* distribution. Finally, Section [4.5](#) summarizes our theoretical findings in this chapter.

4.1 Constrained Gaussian Fields and Harmonic Functions (CGFHF)

CGFHF is a constrained method based on GFHF (ZHU; GHARAMANI; LAFFERTY, 2003). For CGFHF, we maintain the same objective function of GFHF and incorporate the normalization constraints of Liu and Chang (2009) into the optimization problem. Since the proposed optimization problem is convex in $\mathbf{F}$, we found the closed-form solution and describe some theoretical properties. We also provide an effective way to compute CGFHF for transduction and a kernelized approach for this method.

4.1.1 Regularization framework

Mathematically, CGFHF can be viewed as the following constrained optimization problem:

$$\begin{aligned} \min_{\mathbf{F} \in \mathbb{R}^{n \times c}} \quad & \text{tr}(\mathbf{F}^\top \mathbf{L} \mathbf{F}) \\ \text{s.t.} \quad & \mathbf{F}_{\mathcal{L}} = \mathbf{Y}_{\mathcal{L}}, \quad \mathbf{F} \mathbf{1}_c = \mathbf{1}_n, \quad \mathbf{F}^\top \mathbf{1}_n = n\omega \end{aligned} \quad (4.1)$$

As in (2.4), if $\mathbf{F}_{\mathcal{L}} = \mathbf{Y}_{\mathcal{L}}$, then:

$$\text{tr}(\mathbf{F}^\top \mathbf{L} \mathbf{F}) = \text{tr}(\mathbf{Y}_{\mathcal{L}}^\top \mathbf{L}_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}}) + \text{tr}(\mathbf{F}_{\mathcal{U}}^\top \mathbf{L}_{\mathcal{U}} \mathbf{F}_{\mathcal{U}}) + 2\text{tr}(\mathbf{F}_{\mathcal{U}}^\top \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}})$$

From the first constraint, we have:

$$\mathbf{F} \mathbf{1}_c = \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ \mathbf{F}_{\mathcal{U}} \end{bmatrix} \mathbf{1}_c = \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \mathbf{1}_c \\ \mathbf{F}_{\mathcal{U}} \mathbf{1}_c \end{bmatrix} = \begin{bmatrix} \mathbf{1}_l \\ \mathbf{1}_u \end{bmatrix}$$

This yields the following constraint:

$$\mathbf{F}_{\mathcal{U}} \mathbf{1}_c = \mathbf{1}_u$$

From the second constraint, we have:

$$\mathbf{F}^\top \mathbf{1}_n = \begin{bmatrix} \mathbf{Y}_{\mathcal{L}}^\top & \mathbf{F}_{\mathcal{U}}^\top \end{bmatrix} \begin{bmatrix} \mathbf{1}_l \\ \mathbf{1}_u \end{bmatrix} = \mathbf{Y}_{\mathcal{L}}^\top \mathbf{1}_l + \mathbf{F}_{\mathcal{U}}^\top \mathbf{1}_u = n\omega$$

This yields the following constraint:

$$\mathbf{F}_{\mathcal{U}}^\top \mathbf{1}_u = n\omega - \mathbf{Y}_{\mathcal{L}}^\top \mathbf{1}_l$$

Therefore, minimization problem (4.1) can be rewritten as:

$$\begin{aligned} \min_{\mathbf{F}_{\mathcal{U}} \in \mathbb{R}^{u \times c}} \quad & \text{tr} \left(\mathbf{F}_{\mathcal{U}}^{\top} \mathbf{L}_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}} \right) + 2\text{tr} \left(\mathbf{F}_{\mathcal{U}}^{\top} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \right) \\ \text{s.t.} \quad & \mathbf{F}_{\mathcal{U}} \mathbf{1}_c = \mathbf{1}_u, \quad \mathbf{F}_{\mathcal{U}}^{\top} \mathbf{1}_u = n\boldsymbol{\omega} - \mathbf{Y}_{\mathcal{L}}^{\top} \mathbf{1}_l \end{aligned} \quad (4.2)$$

Proposition 4.1 finds the closed-form solution of CGFHF. We used in the proof the simple approach of Lagrange multipliers.

Proposition 4.1 (Solution of CGFHF). *The closed-form solution of (4.2) is given by:*

$$\mathbf{F}_{\mathcal{U}} = -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^{\top} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \boldsymbol{\zeta} + \frac{1}{c} \mathbf{v} \mathbf{1}_c^{\top} \quad (4.3)$$

in which

$$\begin{aligned} \boldsymbol{\zeta} &= n\boldsymbol{\omega}^{\top} - \mathbf{1}_l^{\top} \mathbf{Y}_{\mathcal{L}} + \mathbf{1}_u^{\top} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ \mathbf{v} &= \mathbf{1}_u + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l - \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^{\top} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \mathbf{v}^{(0)} \\ \mathbf{v}^{(0)} &= \mathbf{u} + \mathbf{1}_u^{\top} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l \end{aligned}$$

Proof. The Lagrangian corresponding to (4.2) is given by:

$$\begin{aligned} \mathcal{L}(\mathbf{F}_{\mathcal{U}}, \boldsymbol{\xi}, \boldsymbol{\zeta}) &= \text{tr} \left(\mathbf{F}_{\mathcal{U}}^{\top} \mathbf{L}_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}} \right) + 2\text{tr} \left(\mathbf{F}_{\mathcal{U}}^{\top} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \right) \\ &\quad - \boldsymbol{\xi}^{\top} (\mathbf{F}_{\mathcal{U}} \mathbf{1}_c - \mathbf{1}_u) - \boldsymbol{\zeta}^{\top} (\mathbf{F}_{\mathcal{U}}^{\top} \mathbf{1}_u - n\boldsymbol{\omega} + \mathbf{Y}_{\mathcal{L}}^{\top} \mathbf{1}_l) \end{aligned}$$

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

$$\mathbf{F}_{\mathcal{U}} = -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{1}{2} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \boldsymbol{\xi} \mathbf{1}_c^{\top} + \frac{1}{2} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u \boldsymbol{\zeta}^{\top} \quad (4.4)$$

Substituting (4.4) in the constraint $\mathbf{F}_{\mathcal{U}} \mathbf{1}_c = \mathbf{1}_u$, we obtain:

$$c \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \boldsymbol{\xi} + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u \boldsymbol{\zeta}^{\top} \mathbf{1}_c = 2\mathbf{1}_u + 2\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l \quad (4.5)$$

Substituting (4.4) in the constraint $\mathbf{F}_{\mathcal{U}}^{\top} \mathbf{1}_u = n\boldsymbol{\omega} - \mathbf{Y}_{\mathcal{L}}^{\top} \mathbf{1}_l$, we obtain:

$$\boldsymbol{\zeta}^{\top} = \frac{2}{\mathbf{1}_u^{\top} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \left(n\boldsymbol{\omega}^{\top} - \mathbf{1}_l^{\top} \mathbf{Y}_{\mathcal{L}} + \boldsymbol{\zeta}^{(0)} \right) \quad (4.6)$$

in which

$$\zeta^{(0)} = \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} - \frac{1}{2} \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \xi \mathbf{1}_c^\top$$

Substituting (4.6) in (4.5), we obtain:

$$\xi = \frac{2}{c} \left(\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} - \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1}}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \right)^{-1} \xi^{(0)} \quad (4.7)$$

in which

$$\xi^{(0)} = \mathbf{1}_u + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l - \frac{u \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} - \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}$$

Substituting (4.6) and (4.7) in (4.4), we get (4.3). $\square$

4.1.2 Properties

Corollary 4.2 shows that RMGT is a special case of CGFHF while Corollary 4.3 shows that the constraint $\mathbf{F} \mathbf{1}_c = \mathbf{1}_n$ can be dropped in (4.2) maintaining the same closed-form solution when $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$.

Corollary 4.2 (RMGT as special case of CGFHF). *If $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$, (4.3) is reduced to (2.23).*

Proof. If $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$, then:

$$\mathbf{v} = \mathbf{1}_u + (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{1}_l - \frac{u (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} - \frac{(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u \mathbf{1}_u^\top (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{1}_l}{\mathbf{1}_u^\top (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}$$

We know that $\mathbf{L}_{\mathbb{C}} \mathbf{1}_n = \mathbf{0}_n$. Therefore, we have $(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{1}_l = -(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}} \mathbf{1}_u$. Then, $\mathbf{v} = \mathbf{0}_u$. Substituting $\mathbf{v}$ in (4.3), we get (2.23). $\square$

Corollary 4.3. *If $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$, the solution of (4.2) is equivalent to the solution of the following optimization problem:*

$$\begin{aligned} \min_{\mathbf{F}_{\mathcal{U}} \in \mathbb{R}^{u \times c}} \quad & \text{tr} \left(\mathbf{F}_{\mathcal{U}}^\top (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}} \right) + 2 \text{tr} \left(\mathbf{F}_{\mathcal{U}}^\top (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \right) \\ \text{s.t.} \quad & \mathbf{F}_{\mathcal{U}}^\top \mathbf{1}_u = n\omega - \mathbf{Y}_{\mathcal{L}}^\top \mathbf{1}_l \end{aligned}$$

Proof. The proof comes directly from Corollary 4.2 and the steps shown in Proposition 4.1. $\square$

4.1.3 Computing CGFHF

Let $\mathbf{M} := -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1}\mathbf{L}_{\mathcal{U}\mathcal{L}}\mathbf{Y}_{\mathcal{L}} \in \mathbb{R}^{u \times c}$. Therefore, the vector $\boldsymbol{\zeta} \in \mathbb{R}^{1 \times c}$ in (4.3) can be easily computed as:

$$\boldsymbol{\zeta} = n\boldsymbol{\omega}^\top - \mathbf{1}_l^\top \mathbf{Y}_{\mathcal{L}} - \mathbf{1}_u^\top \mathbf{M} \quad (4.8)$$

Therefore, $\forall i \in \mathbb{N}_c$, we have:

$$\zeta_i = n\omega_i - |\mathcal{C}_i| - \sum_{j=1}^u \mathbf{M}_{ji}$$

The vector $\mathbf{v} \in \mathbb{R}^u$ in (4.3) can be computed as:

$$\mathbf{v} = \mathbf{1}_u - \mathbf{M}\mathbf{1}_c - \psi \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u$$

in which

$$\psi = \frac{u - \mathbf{1}_u^\top \mathbf{M}\mathbf{1}_c}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} = \frac{u - \sum_{i=1}^n \sum_{j=1}^c \mathbf{M}_{ij}}{\sum_{i=1}^n \sum_{j=1}^c (\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1})_{ij}}$$

Therefore, we have:

$$v_i = 1 - \sum_{j=1}^c \mathbf{M}_{ij} - \psi \sum_{j=1}^n (\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1})_{ij}$$

Remark 4.4. Given $\mathbf{a} \in \mathbb{R}^s$ and $\mathbf{b} \in \mathbb{R}^r$, with $r, s \in \mathbb{N}^*$, we have:

1. $\mathbf{B} = \mathbf{b}\mathbf{1}_s^\top$ is computed as $\mathbf{B}_{\cdot,j} = \mathbf{b}$, $\forall j \in \mathbb{N}_s$;
2. $\bar{\mathbf{B}} = \mathbf{1}_s \mathbf{b}^\top$ is computed as $\bar{\mathbf{B}}_{i,\cdot} = \mathbf{b}^\top$, $\forall i \in \mathbb{N}_s$;
3. $\mathbf{A} = \mathbf{a}\mathbf{b}^\top$ is computed as $\mathbf{A}_{ij} = \mathbf{a}_i \mathbf{b}_j$, $\forall i \in \mathbb{N}_r, \forall j \in \mathbb{N}_s$.

From Remark 4.4, we can rewrite (4.3) as:

$$(\mathbf{F}_{\mathcal{U}})_{ij} = \mathbf{M}_{ij} + \frac{\zeta_j \sum_{p=1}^u (\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1})_{ip}}{\sum_{p=1}^u \sum_{q=1}^u (\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1})_{pq}} + \frac{1}{c} v_i \quad (4.9)$$

From (4.9), we obtain Corollary 4.5, which describes special cases of (4.9) for transduction. These cases are summarized in Alg. 1.

Corollary 4.5 (Transductive classification using CGFHF). From (4.9), the output class of $\mathbf{x}_i \in \mathcal{X}$ is given by:

$$y(\mathbf{x}_i) = \arg \max_{j \in \mathbb{N}_c} (\mathbf{F}_{\mathcal{U}})_{ij} = \arg \max_{j \in \mathbb{N}_c} \mathbf{M}_{ij} + \frac{\zeta_j \sum_{p=1}^n (\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1})_{ip}}{\sum_{p=1}^n \sum_{q=1}^n (\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1})_{pq}} \quad (4.10)$$

Moreover, if we also assume that $\omega = \mathbf{1}_c/c$, then:

$$y(\mathbf{x}_i) = \arg \max_{j \in \mathbb{N}_c} \mathbf{M}_{ij} - \frac{\left(\sum_{p=1}^n \mathbf{M}_{pj} \right) \sum_{p=1}^n (\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1})_{ip}}{\sum_{p=1}^n \sum_{q=1}^n (\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1})_{pq}} \quad (4.11)$$

Proof. Since $\mathbf{v}_i$ in (4.9) does not depend on class j , we get (4.10). From (4.10), if $\omega = \mathbf{1}_c/c$, we obtain:

$$y(\mathbf{x}_i) = \arg \max_{j \in \mathbb{N}_c} \mathbf{M}_{ij} + \frac{\left(\frac{n}{c} - |\mathcal{C}_j| - \sum_{p=1}^n \mathbf{M}_{pj} \right) \sum_{p=1}^n (\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1})_{ip}}{\sum_{p=1}^n \sum_{q=1}^n (\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1})_{pq}}$$

This is equivalent to (4.11). □

Alg. 1 summarizes our theoretical analysis for CGFHF.

Algorithm 1 – CGFHF for transduction

Input: a graph Laplacian $\mathbf{L} \in \mathbb{R}^{n \times n}$ and a label matrix $\mathbf{Y}_{\mathcal{L}} \in \mathbb{B}^{l \times c}$

1. Compute $\mathbf{M} = -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}$
2. Compute $\zeta \in \mathbb{R}^{1 \times c}$ as in (4.8)
3. Compute the class labels as in (4.10) or in (4.11)

Output: the class labels $y(\mathbf{x}_i)$, $i \in \mathbb{N}_n$

4.2 Constrained Local and Global Consistency (CLGC)

CLGC is a constrained method based on LGC (ZHOU *et al.*, 2004). For CLGC, we maintain the same objective function of LGC and incorporate the normalization constraints of Liu and Chang (2009) into the optimization problem. Since the proposed optimization problem is convex in $\mathbf{F}$, we found the closed-form solution and describe some theoretical properties. We also provide an effective way to compute CLGC for transduction.

4.2.1 Regularization framework

Mathematically, CLGC can be viewed as the following constrained optimization problem:

$$\begin{aligned} \min_{\mathbf{F} \in \mathbb{R}^{n \times c}} \quad & \text{tr} \left(\mathbf{F}^\top \mathbf{L} \mathbf{F} + (\mathbf{F} - \mathbf{Y})^\top \Sigma (\mathbf{F} - \mathbf{Y}) \right) \\ \text{s.t.} \quad & \mathbf{F} \mathbf{1}_c = \mathbf{1}_n, \quad \mathbf{F}^\top \mathbf{1}_n = n\omega \end{aligned} \quad (4.12)$$

Proposition 4.6 finds the closed-form solution of CLGC applying the Lagrange multipliers.

Proposition 4.6 (Solution of CLGC). *The closed-form solution of (4.12) is given by:*

$$\mathbf{F} = \Theta \Sigma \mathbf{Y} + \frac{\Theta \mathbf{1}_n}{\mathbf{1}_n^\top \Theta \mathbf{1}_n} \zeta + \frac{1}{c} \mathbf{v} \mathbf{1}_c^\top \quad (4.13)$$

where

$$\begin{aligned} \Theta &= (\mathbf{L} + \Sigma)^{-1} \\ \zeta &= n\omega^\top - \mathbf{1}_n^\top \Theta \Sigma \mathbf{Y} \\ \mathbf{v} &= \mathbf{1}_n - \Theta \Sigma \mathbf{Y} \mathbf{1}_c - \frac{\Theta \mathbf{1}_n}{\mathbf{1}_n^\top \Theta \mathbf{1}_n} \left(n - \mathbf{1}_n^\top \Theta \Sigma \mathbf{Y} \mathbf{1}_c \right) \end{aligned}$$

Proof. The Lagrangian corresponding to (4.12) is given by:

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

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

$$\mathbf{F} = \Theta \left(\Sigma \mathbf{Y} + \frac{1}{2} \xi \mathbf{1}_c^\top + \frac{1}{2} \mathbf{1}_n \lambda^\top \right) \quad (4.14)$$

where $\Theta = (\mathbf{L} + \Sigma)^{-1}$. Substituting (4.14) in the constraint $\mathbf{F} \mathbf{1}_c = \mathbf{1}_n$, we obtain:

$$\Theta \Sigma \mathbf{Y} \mathbf{1}_c + \frac{c}{2} \Theta \xi + \frac{1}{2} \Theta \mathbf{1}_n \lambda^\top \mathbf{1}_c = \mathbf{1}_n \quad (4.15)$$

Substituting (4.14) in the constraint $\mathbf{F}^\top \mathbf{1}_n = n\omega$, we obtain:

$$\lambda^\top = \frac{2}{\mathbf{1}_n^\top \Theta \mathbf{1}_n} \left(n\omega^\top - \mathbf{1}_n^\top \Theta \Sigma \mathbf{Y} - \frac{1}{2} \mathbf{1}_n^\top \Theta \xi \mathbf{1}_c^\top \right) \quad (4.16)$$

Substituting (4.16) in (4.15), we obtain:

$$\xi = \frac{2}{c} \left(\Theta - \frac{\Theta \mathbf{1}_n \mathbf{1}_n^\top \Theta}{\mathbf{1}_n^\top \Theta \mathbf{1}_n} \right)^{-1} \left(\mathbf{1}_n - \Theta \Sigma \mathbf{Y} \mathbf{1}_c - \frac{\Theta \mathbf{1}_n}{\mathbf{1}_n^\top \Theta \mathbf{1}_n} \left(n - \mathbf{1}_n^\top \Theta \Sigma \mathbf{Y} \mathbf{1}_c \right) \right) \quad (4.17)$$

Substituting (4.16) in (4.14), we obtain:

$$\mathbf{F} = \Theta \Sigma \mathbf{Y} + \frac{\Theta \mathbf{1}_n}{\mathbf{1}_n^\top \Theta \mathbf{1}_n} \zeta + \frac{1}{2} \varpi \xi \mathbf{1}_c^\top \quad (4.18)$$

in which

$$\begin{aligned} \zeta &= n \omega^\top - \mathbf{1}_n^\top \Theta \Sigma \mathbf{Y} \\ \varpi &= \Theta \left(\mathbf{I}_n - \frac{\mathbf{1}_n \mathbf{1}_n^\top \Theta}{\mathbf{1}_n^\top \Theta \mathbf{1}_n} \right) \end{aligned}$$

Substituting (4.17) in (4.18), we obtain (4.13). $\square$

4.2.2 Properties

Proposition 4.7 shows that CLGC generalizes CGFHF. Moreover, Corollary 4.8 shows that CLGC also generalizes RMGT. Specifically, these results show that RMGT and CGFHF are special cases of CLGC.

Proposition 4.7 (CGFHF as special case of CLGC). *If $\Sigma_{\mathcal{U}} = \mathbf{O}_{u \times u}$ and $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$, (4.13) is reduced to (4.3), approximately.*

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

$$\begin{bmatrix} \mathbf{L}_{\mathcal{L}\mathcal{L}} + \Sigma_{\mathcal{L}} & \mathbf{L}_{\mathcal{L}\mathcal{U}} \\ \mathbf{L}_{\mathcal{U}\mathcal{L}} & \mathbf{L}_{\mathcal{U}\mathcal{U}} \end{bmatrix} \begin{bmatrix} \Theta_{\mathcal{L}\mathcal{L}} & \Theta_{\mathcal{L}\mathcal{U}} \\ \Theta_{\mathcal{U}\mathcal{L}} & \Theta_{\mathcal{U}\mathcal{U}} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_l & \mathbf{O}_{l \times u} \\ \mathbf{O}_{u \times l} & \mathbf{I}_u \end{bmatrix}$$

This yields:

$$\begin{aligned} \Theta_{\mathcal{L}\mathcal{L}} &= \Gamma^{-1} \Sigma_{\mathcal{L}}^{-1} \\ \Theta_{\mathcal{U}\mathcal{L}} &= -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \Gamma^{-1} \Sigma_{\mathcal{L}}^{-1} \\ \Theta_{\mathcal{L}\mathcal{U}} &= -\Gamma^{-1} \Sigma_{\mathcal{L}}^{-1} \mathbf{L}_{\mathcal{L}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \\ \Theta_{\mathcal{U}\mathcal{U}} &= \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \Gamma^{-1} \Sigma_{\mathcal{L}}^{-1} \mathbf{L}_{\mathcal{L}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \end{aligned} \quad (4.19)$$

in which

$$\Gamma = \Sigma_{\mathcal{L}}^{-1} \mathbf{L}_{\mathcal{L}\mathcal{L}} + \mathbf{I}_l - \Sigma_{\mathcal{L}}^{-1} \mathbf{L}_{\mathcal{L}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}}$$

Therefore, (4.13) can be rewritten as:

$$\begin{aligned} \mathbf{F} = & \begin{bmatrix} \Theta_{\mathcal{L}\mathcal{L}}\Sigma_{\mathcal{L}}\mathbf{Y}_{\mathcal{L}} \\ \Theta_{\mathcal{U}\mathcal{L}}\Sigma_{\mathcal{L}}\mathbf{Y}_{\mathcal{L}} \end{bmatrix} + \beta^{(0)}\beta^{(1)} \\ & + \frac{1}{c} \left\{ \begin{bmatrix} \mathbf{1}_l - \Theta_{\mathcal{L}\mathcal{L}}\Sigma_{\mathcal{L}}\mathbf{1}_l \\ \mathbf{1}_u - \Theta_{\mathcal{U}\mathcal{L}}\Sigma_{\mathcal{L}}\mathbf{1}_l \end{bmatrix} - \beta^{(0)}\beta^{(2)} \right\} \mathbf{1}_c^\top \end{aligned} \quad (4.20)$$

in which

$$\begin{aligned} \beta^{(0)} &= \frac{1}{\tau} \begin{bmatrix} \Theta_{\mathcal{L}\mathcal{L}}\mathbf{1}_l + \Theta_{\mathcal{L}\mathcal{U}}\mathbf{1}_u \\ \Theta_{\mathcal{U}\mathcal{L}}\mathbf{1}_l + \Theta_{\mathcal{U}\mathcal{U}}\mathbf{1}_u \end{bmatrix} \\ \beta^{(1)} &= n\omega^\top - \mathbf{1}_l^\top \Theta_{\mathcal{L}\mathcal{L}}\Sigma_{\mathcal{L}}\mathbf{Y}_{\mathcal{L}} - \mathbf{1}_u^\top \Theta_{\mathcal{U}\mathcal{L}}\Sigma_{\mathcal{L}}\mathbf{Y}_{\mathcal{L}} \\ \beta^{(2)} &= n - \mathbf{1}_l^\top \Theta_{\mathcal{L}\mathcal{L}}\Sigma_{\mathcal{L}}\mathbf{1}_l - \mathbf{1}_u^\top \Theta_{\mathcal{U}\mathcal{L}}\Sigma_{\mathcal{L}}\mathbf{1}_l \end{aligned} \quad (4.21)$$

such that

$$\tau = \mathbf{1}_l^\top \Theta_{\mathcal{L}\mathcal{L}}\mathbf{1}_l + \mathbf{1}_u^\top \Theta_{\mathcal{U}\mathcal{L}}\mathbf{1}_l + \mathbf{1}_l^\top \Theta_{\mathcal{L}\mathcal{U}}\mathbf{1}_u + \mathbf{1}_u^\top \Theta_{\mathcal{U}\mathcal{U}}\mathbf{1}_u$$

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

$$\begin{aligned} \Theta_{\mathcal{L}\mathcal{U}} &\approx \mathbf{O}_{l \times u}, & \Theta_{\mathcal{U}\mathcal{U}} &\approx \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1}, & \Theta_{\mathcal{L}\mathcal{L}}\Sigma_{\mathcal{L}} &\approx \mathbf{I}_l, \\ \Theta_{\mathcal{L}\mathcal{L}} &\approx \mathbf{O}_{l \times l}, & \Theta_{\mathcal{U}\mathcal{L}}\Sigma_{\mathcal{L}} &= -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1}\mathbf{L}_{\mathcal{U}\mathcal{L}}, \\ \Theta_{\mathcal{U}\mathcal{L}} &\approx \mathbf{O}_{u \times l} \end{aligned} \quad (4.22)$$

Substituting the approximations in (4.22) in (4.21), we obtain:

$$\begin{aligned} \beta^{(0)} &\approx \frac{1}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \begin{bmatrix} \mathbf{0}_l \\ \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u \end{bmatrix} \\ \beta^{(1)} &\approx n\omega^\top - \mathbf{1}_l^\top \mathbf{Y}_{\mathcal{L}} + \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ \beta^{(2)} &\approx u + \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l \end{aligned} \quad (4.23)$$

Substituting $\beta^{(0)}$, $\beta^{(1)}$, and $\beta^{(2)}$ in (4.20), we obtain (4.3), approximately. $\square$

Corollary 4.8 (RMGT as special case of CLGC). Consider $\Sigma_{\mathcal{U}} = \mathbf{O}_{u \times u}$ and $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$. If we also have $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$, (4.13) is reduced to (2.23), approximately.

Proof. From Proposition 4.7, Eq. (4.13) is reduced to (4.3), approximately, if $\Sigma_{\mathcal{U}} = \mathbf{O}_{u \times u}$ and $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$. From Corollary 4.2, (4.3) is reduced to (2.23) if $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$. $\square$

4.2.3 Computing CLGC

Let $\mathbf{M} := \Theta \Sigma \mathbf{Y} \in \mathbb{R}^{n \times c}$. By definition, we can write:

$$\mathbf{M} = \begin{bmatrix} \Theta_{\mathcal{L}\mathcal{L}} & \Theta_{\mathcal{L}\mathcal{U}} \\ \Theta_{\mathcal{U}\mathcal{L}} & \Theta_{\mathcal{U}\mathcal{U}} \end{bmatrix} \begin{bmatrix} \Sigma_{\mathcal{L}} & \mathbf{O}_{l \times u} \\ \mathbf{O}_{u \times l} & \Sigma_{\mathcal{U}} \end{bmatrix} \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ \mathbf{O}_{u \times c} \end{bmatrix} = \begin{bmatrix} \Theta_{\mathcal{L}\mathcal{L}} \Sigma_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ \Theta_{\mathcal{U}\mathcal{L}} \Sigma_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \end{bmatrix}$$

Note that $\mathbf{M}$ has no dependency on $\Sigma_{\mathcal{U}}$. This means that the values of the regularization parameters on the unlabeled examples have no influence on the computation of $\mathbf{M}$. In order to compute $\mathbf{M}$ efficiently, we take into account Remark 4.9 and Algorithm 2.

Remark 4.9. Let $\mathbf{A} \in \mathbb{R}^{a \times b}$ and $\mathbf{B} \in \mathbb{R}^{b \times c}$ such that $\mathbf{A}$ is a diagonal matrix. The matrix $\mathbf{C} := \mathbf{AB} \in \mathbb{R}^{a \times c}$ can be computed as:

$$\mathbf{C}_{ij} = \sum_{k=1}^b \mathbf{A}_{ik} \mathbf{B}_{kj} = \mathbf{A}_{ii} \mathbf{B}_{ij}$$

$\forall i \in \mathbb{N}_a, \forall j \in \mathbb{N}_c$. If $\mathbf{B}$ is a binary matrix such that $\mathbf{B} \mathbf{1}_c = \mathbf{1}_b$ and $\mathbf{v} \in \mathbb{N}_c^b$ such that $\mathbf{v}_i = \arg \max_{j \in \mathbb{N}_c} \mathbf{B}_{ij}$, $\forall i \in \mathbb{N}_b$, we have $\mathbf{C}_{ij} = \mathbf{A}_{ii}$ (with $i \in \mathbb{N}_a$ and $j \in \mathbb{N}_c$) if $\mathbf{v}_i = j$ and 0 otherwise. Moreover, if $\mathbf{X} := \mathbf{DC} \in \mathbb{R}^{d \times c}$ for a given matrix $\mathbf{D} \in \mathbb{R}^{d \times a}$, we have:

$$\mathbf{X}_{ij} = \sum_{k=1}^a \mathbf{D}_{ik} \mathbf{C}_{kj} = \sum_{\mathbf{v}_k=j} \mathbf{D}_{ik} \mathbf{A}_{kk}$$

$$\forall i \in \mathbb{N}_d, \forall j \in \mathbb{N}_c.$$

Proof. The proof comes directly from the matrix product definition. □

Alg. 2 summarizes how we efficiently compute the matrix product $\mathbf{X} = \mathbf{DC}$, as defined in Remark 4.9. This simple approach minimizes the number of floating points multiplications required to compute the corresponding matrix product.

Algorithm 2 – Computing $\mathbf{X} = \mathbf{DC}$

Input: matrices $\mathbf{A} \in \mathbb{R}^{a \times b}$, $\mathbf{B} \in \mathbb{R}^{b \times c}$, $\mathbf{C} = \mathbf{AB} \in \mathbb{R}^{a \times c}$, and $\mathbf{D} \in \mathbb{R}^{d \times a}$, as described in Remark 4.9

$\mathbf{X} \leftarrow \mathbf{O}_{d \times c}$

for $i \leftarrow 1$ **to** n **do**

$j \leftarrow \mathbf{v}_i$

for $k \leftarrow 1$ **to** d **do**

$\mathbf{X}_{kj} \leftarrow \mathbf{X}_{kj} + \mathbf{D}_{ki} \mathbf{A}_{ii}$

end for

end for

Output: matrix $\mathbf{X}$

Since $\Sigma_{\mathcal{L}}$ is a diagonal matrix and $\mathbf{Y}_{\mathcal{L}}$ is a binary matrix such that $\mathbf{Y}_{\mathcal{L}}\mathbf{1}_c = \mathbf{1}_l$, we can use Alg. 2 to compute the matrix products $\Theta_{\mathcal{L}\mathcal{L}}\Sigma_{\mathcal{L}}\mathbf{Y}_{\mathcal{L}}$ and $\Theta_{\mathcal{U}\mathcal{L}}\Sigma_{\mathcal{L}}\mathbf{Y}_{\mathcal{L}}$. Therefore, $\mathbf{M}$ can be computed in an efficient way by using Alg. 2, making $\mathbf{A} = \Sigma_{\mathcal{L}}$, $\mathbf{B} = \mathbf{Y}_{\mathcal{L}}$, and $\mathbf{D} = \Theta_{\mathcal{L}\mathcal{L}}$ or $\mathbf{D} = \Theta_{\mathcal{U}\mathcal{L}}$. Mathematically, we have:

$$\mathbf{M}_{ij} = \begin{cases} \sum_{y_k=j} (\Theta_{\mathcal{L}\mathcal{L}})_{ik} (\Sigma_{\mathcal{L}})_{kk} & , i \leq l \\ \sum_{y_k=j} (\Theta_{\mathcal{U}\mathcal{L}})_{ik} (\Sigma_{\mathcal{L}})_{kk} & , l < i \leq n \end{cases} \quad (4.24)$$

The vector $\zeta \in \mathbb{R}^{1 \times c}$ in (4.13) can be easily computed as:

$$\zeta = n\omega^\top - \mathbf{1}_n^\top \mathbf{M} \quad (4.25)$$

Therefore, $\forall i \in \mathbb{N}_c$, we have:

$$\zeta_i = n\omega_i - \sum_{j=1}^n \mathbf{M}_{ji}$$

The vector $\mathbf{v} \in \mathbb{R}^n$ in (4.13) can be computed as:

$$\mathbf{v} = \mathbf{1}_n - \mathbf{M}\mathbf{1}_c - \psi\Theta\mathbf{1}_n$$

in which

$$\psi = \frac{n - \mathbf{1}_n^\top \mathbf{M}\mathbf{1}_c}{\mathbf{1}_n^\top \Theta \mathbf{1}_n} = \frac{n - \sum_{i=1}^n \sum_{j=1}^c \mathbf{M}_{ij}}{\sum_{i=1}^n \sum_{j=1}^n \Theta_{ij}}$$

Therefore, $\forall i \in \mathbb{N}_n$, we have:

$$\mathbf{v}_i = 1 - \sum_{j=1}^c \mathbf{M}_{ij} - \psi \sum_{j=1}^n \Theta_{ij}$$

From Remark 4.4, we can rewrite (4.13) as:

$$\mathbf{F}_{ij} = \mathbf{M}_{ij} + \frac{\zeta_j \sum_{p=1}^n \Theta_{ip}}{\sum_{p=1}^n \sum_{q=1}^n \Theta_{pq}} + \frac{1}{c} \mathbf{v}_i \quad (4.26)$$

From (4.26), we obtain Corollary 4.10, which describes special cases of (4.26) for transduction.

Corollary 4.10 (Transductive classification using CLGC). *From (4.26), the output class of $\mathbf{x}_i \in \mathcal{X}$ is given by:*

$$y(\mathbf{x}_i) = \arg \max_{j \in \mathbb{N}_c} \mathbf{F}_{ij} = \arg \max_{j \in \mathbb{N}_c} \mathbf{M}_{ij} + \frac{\zeta_j \sum_{p=1}^n \Theta_{ip}}{\sum_{p=1}^n \sum_{q=1}^n \Theta_{pq}} \quad (4.27)$$

Moreover, if we also assume that $\omega = \mathbf{1}_c/c$, then:

$$y(\mathbf{x}_i) = \arg \max_{j \in \mathbb{N}_c} \mathbf{M}_{ij} - \frac{\left(\sum_{p=1}^n \mathbf{M}_{pj} \right) \sum_{p=1}^n \Theta_{ip}}{\sum_{p=1}^n \sum_{q=1}^n \Theta_{pq}} \quad (4.28)$$

Proof. Since v_i in (4.26) does not depend on class j , we get (4.27). Moreover, if we also assume that $\omega_i = 1/c, \forall i \in \mathbb{N}_c$, we obtain:

$$y(\mathbf{x}_i) = \arg \max_{j \in \mathbb{N}_c} \mathbf{M}_{ij} + \frac{\left(n/c - \sum_{p=1}^n \mathbf{M}_{pj} \right) \sum_{p=1}^n \Theta_{ip}}{\sum_{p=1}^n \sum_{q=1}^n \Theta_{pq}}$$

This is equivalent to (4.28). □

Alg. 3 summarizes our theoretical analysis of CLGC for transduction.

Algorithm 3 – CLGC for transduction

Input: a graph Laplacian $\mathbf{L} \in \mathbb{R}^{n \times n}$, a nonnegative diagonal matrix $\Sigma \in \mathbb{R}^{n \times n}$, and a label matrix $\mathbf{Y} \in \mathbb{B}^{n \times c}$

1. Compute $\Theta = (\mathbf{L} + \Sigma)^{-1} \in \mathbb{R}^{n \times n}$
2. Compute $\mathbf{M} \in \mathbb{R}^{n \times c}$ as in (4.24)
3. Compute $\zeta \in \mathbb{R}^{1 \times c}$ as in (4.25)
4. Compute the class labels as in (4.27) or in (4.28)

Output: the class labels $y(\mathbf{x}_i), i \in \mathbb{N}_n$

4.3 Constrained Laplacian Regularized Least Squares (CLapRLS)

CLapRLS is a constrained method based on LapRLS (BELKIN; NIYOGI; SINDHWANI, 2006). For CLapRLS, we maintain the same objective function of LapRLS and incorporate the normalization constraints of Liu and Chang (2009) into the optimization problem. Since the proposed optimization problem is convex in $\mathbf{F}$, we found the closed-form solution and describe some theoretical properties. We also provide an effective way to compute CLapRLS for induction.

4.3.1 Regularization framework

Through the kernel expansions in (BELKIN; NIYOGI; SINDHWANI, 2006), we have $\mathbf{F} = \mathbf{K}\alpha$, in which $\alpha \in \mathbb{R}^{n \times c}$. Therefore, the normalization constraints of Liu and Chang (2009) can be rewritten as:

$$\mathbf{K}\alpha\mathbf{1}_c = \mathbf{1}_n, \quad \alpha^\top \mathbf{K}\mathbf{1}_n = n\omega$$

Mathematically, CLapRLS can be viewed as the following constrained optimization problem:

$$\begin{aligned} \min_{\alpha \in \mathbb{R}^{n \times c}} \quad & \text{tr} \left((\mathbf{K}\alpha - \mathbf{Y})^\top \Sigma (\mathbf{K}\alpha - \mathbf{Y}) + \gamma_A \alpha^\top \mathbf{K}\alpha + \gamma_l \alpha^\top \mathbf{K}\mathbf{L}\mathbf{K}\alpha \right) \\ \text{s.t.} \quad & \mathbf{K}\alpha\mathbf{1}_c = \mathbf{1}_n, \quad \alpha^\top \mathbf{K}\mathbf{1}_n = n\omega \end{aligned} \quad (4.29)$$

Proposition 4.11 finds the closed-form solution of CLapRLS applying the Lagrange multipliers.

Proposition 4.11 (Solution of CLapRLS). *The closed-form solution of (4.29) is given by:*

$$\alpha = \Theta \Sigma \mathbf{Y} + \frac{\Theta \mathbf{1}_n}{\mathbf{1}_n^\top \mathbf{K} \Theta \mathbf{1}_n} \zeta + \frac{1}{c} \mathbf{v} \mathbf{1}_c^\top \quad (4.30)$$

in which

$$\begin{aligned} \Theta &= (\Sigma \mathbf{K} + \gamma_A \mathbf{I}_n + \gamma_l \mathbf{L}\mathbf{K})^{-1} \\ \zeta &= n\omega^\top - \mathbf{1}_n^\top \mathbf{K} \Theta \Sigma \mathbf{Y} \\ \mathbf{v} &= \mathbf{K}^{-1} \mathbf{1}_n - \Theta \Sigma \mathbf{Y} \mathbf{1}_c - \frac{\Theta \mathbf{1}_n}{\mathbf{1}_n^\top \mathbf{K} \Theta \mathbf{1}_n} \mathbf{v}^{(0)} \\ \mathbf{v}^{(0)} &= n - \mathbf{1}_n^\top \mathbf{K} \Theta \Sigma \mathbf{Y} \mathbf{1}_c \end{aligned} \quad (4.31)$$

Therefore, the transductive solution is given by:

$$\mathbf{F} = \mathbf{K} \Theta \Sigma \mathbf{Y} + \frac{\mathbf{K} \Theta \mathbf{1}_n}{\mathbf{1}_n^\top \mathbf{K} \Theta \mathbf{1}_n} \zeta + \frac{1}{c} \mathbf{K} \mathbf{v} \mathbf{1}_c^\top \quad (4.32)$$

Proof. The Lagrangian corresponding to (4.29) is given by:

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

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

$$\alpha = \Theta \left(\Sigma \mathbf{Y} + \frac{1}{2} \xi \mathbf{1}_c^\top + \frac{1}{2} \mathbf{1}_n \lambda^\top \right) \quad (4.33)$$

Substituting (4.33) in $\mathbf{K}\alpha\mathbf{1}_c = \mathbf{1}_n$, we obtain:

$$\mathbf{K} \Theta \Sigma \mathbf{Y} \mathbf{1}_c + \frac{c}{2} \mathbf{K} \Theta \xi + \frac{1}{2} \mathbf{K} \Theta \mathbf{1}_n \lambda^\top \mathbf{1}_c = \mathbf{1}_n \quad (4.34)$$

Substituting (4.33) in $\alpha^\top \mathbf{K} \mathbf{1}_n = n\omega$, we obtain:

$$\lambda^\top = \frac{2}{\mathbf{1}_n^\top \mathbf{K} \Theta \mathbf{1}_n} \left(n\omega^\top - \mathbf{1}_n^\top \mathbf{K} \Theta \Sigma \mathbf{Y} - \frac{1}{2} \mathbf{1}_n^\top \mathbf{K} \Theta \xi \mathbf{1}_c^\top \right) \quad (4.35)$$

Substituting (4.35) in (4.34), we obtain:

$$\xi = \frac{2}{c} \left(\mathbf{K} \Theta - \frac{\mathbf{K} \Theta \mathbf{1}_n \mathbf{1}_n^\top \mathbf{K} \Theta}{\mathbf{1}_n^\top \mathbf{K} \Theta \mathbf{1}_n} \right)^{-1} \xi^{(0)} \quad (4.36)$$

in which

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

Substituting (4.35) in (4.33), we obtain:

$$\alpha = \Theta \Sigma \mathbf{Y} + \frac{\Theta \mathbf{1}_n}{\mathbf{1}_n^\top \mathbf{K} \Theta \mathbf{1}_n} \left(n\omega^\top - \mathbf{1}_n^\top \mathbf{K} \Theta \Sigma \mathbf{Y} \right) + \frac{1}{2} \Theta \left(\mathbf{I}_n - \frac{\mathbf{1}_n \mathbf{1}_n^\top \mathbf{K} \Theta}{\mathbf{1}_n^\top \mathbf{K} \Theta \mathbf{1}_n} \right) \xi \mathbf{1}_c^\top \quad (4.37)$$

Substituting (4.36) in (4.37) yields (4.30). Applying $\mathbf{F} = \mathbf{K} \alpha$ yields (4.32). $\square$

4.3.2 Properties

Corollaries 4.12, 4.13, and 4.14 show the inductive forms of CLGC, CGFHF, and RMGT, respectively. Specifically, these corollaries show that CLGC, CGFHF, and RMGT are inductive methods, instead of transductive ones. Moreover, this means that they are special cases of CLapRLS.

Corollary 4.12 (*Inductive form of CLGC*). If $\gamma_A = 0$ and $\gamma_I = 1$, (4.30) is reduced to:

$$\alpha = \mathbf{K}^{-1} \left(\Theta \Sigma \mathbf{Y} + \frac{\Theta \mathbf{1}_n}{\mathbf{1}_n^\top \Theta \mathbf{1}_n} \zeta + \frac{1}{c} \mathbf{v} \mathbf{1}_c^\top \right) \quad (4.38)$$

where

$$\begin{aligned} \Theta &= (\mathbf{L} + \Sigma)^{-1} \\ \zeta &= n\omega^\top - \mathbf{1}_n^\top \Theta \Sigma \mathbf{Y} \\ \mathbf{v} &= \mathbf{1}_n - \Theta \Sigma \mathbf{Y} \mathbf{1}_c - \frac{\Theta \mathbf{1}_n}{\mathbf{1}_n^\top \Theta \mathbf{1}_n} \left(n - \mathbf{1}_n^\top \Theta \Sigma \mathbf{Y} \mathbf{1}_c \right) \end{aligned}$$

Therefore, the transductive solution is given by:

$$\mathbf{F} = \Theta \Sigma \mathbf{Y} + \frac{\Theta \mathbf{1}_n}{\mathbf{1}_n^\top \Theta \mathbf{1}_n} \zeta + \frac{1}{c} \mathbf{v} \mathbf{1}_c^\top \quad (4.39)$$

Proof. Substitute $\gamma_A = 0$ and $\gamma_l = 1$ in (4.30) and (4.32) to get (4.38) and (4.39), respectively. $\square$

Corollary 4.13 (Inductive form of CGFHF). *If $\Sigma_{\mathcal{U}} = \mathbf{O}_{u \times u}$ and $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$, (4.38) is reduced to:*

$$\alpha \approx \mathbf{K}^{-1} \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \zeta + \frac{1}{c} \mathbf{v} \mathbf{1}_c^\top \end{bmatrix} \quad (4.40)$$

in which

$$\begin{aligned} \zeta &= n\omega^\top - \mathbf{1}_l^\top \mathbf{Y}_{\mathcal{L}} + \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ \mathbf{v} &= \mathbf{1}_u + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l - \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \mathbf{v}^{(0)} \\ \mathbf{v}^{(0)} &= u + \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l \end{aligned}$$

Therefore, the transductive solution is given by:

$$\mathbf{F} = \mathbf{K}\alpha \approx \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \zeta + \frac{1}{c} \mathbf{v} \mathbf{1}_c^\top \end{bmatrix} \quad (4.41)$$

Proof. Applying Proposition 4.7 in (4.38) yields (4.40). From (4.40), we get (4.41). $\square$

Corollary 4.14 (Inductive form of RMGT). *If $\mathbf{L} = \mathbf{L}_{\mathbf{C}}$, (4.40) is reduced to:*

$$\alpha \approx \mathbf{K}^{-1} \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -(\mathbf{L}_{\mathbf{C}})_{\mathcal{U}\mathcal{U}}^{-1} (\mathbf{L}_{\mathbf{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{(\mathbf{L}_{\mathbf{C}})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top (\mathbf{L}_{\mathbf{C}})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \zeta \end{bmatrix} \quad (4.42)$$

in which

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

Therefore, the transductive solution is given by:

$$\mathbf{F} = \mathbf{K}\alpha \approx \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -(\mathbf{L}_{\mathbf{C}})_{\mathcal{U}\mathcal{U}}^{-1} (\mathbf{L}_{\mathbf{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{(\mathbf{L}_{\mathbf{C}})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top (\mathbf{L}_{\mathbf{C}})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \zeta \end{bmatrix} \quad (4.43)$$

Proof. The proof comes directly from Corollary 4.2. $\square$

4.3.3 Computing CLapRLS

Assume the same matrix $\mathbf{M} = \Theta \Sigma \mathbf{Y}$ defined in Section 4.2 such that Θ is defined in (4.31). Therefore, $\mathbf{M}_{ij}$ can be computed as in (4.24). The vector $\zeta \in \mathbb{R}^{1 \times c}$ in (4.31) can be easily computed as:

$$\zeta = n\omega^\top - \mathbf{1}_n^\top \mathbf{K} \mathbf{M} \quad (4.44)$$

Therefore, $\forall i \in \mathbb{N}_c$, we have:

$$\zeta_i = n\omega_i - \sum_{z=1}^n (\mathbf{K} \mathbf{M})_{zi}$$

The vector $\mathbf{v} \in \mathbb{R}^n$ in (4.31) can be computed as:

$$\mathbf{v} = \mathbf{K}^{-1} \mathbf{1}_n - \mathbf{M} \mathbf{1}_c - \psi \Theta \mathbf{1}_n$$

in which

$$\psi = \frac{n - \mathbf{1}_n^\top \mathbf{K} \mathbf{M} \mathbf{1}_c}{\mathbf{1}_n^\top \mathbf{K} \Theta \mathbf{1}_n} = \frac{n - \sum_{i=1}^n \sum_{j=1}^c (\mathbf{K} \mathbf{M})_{ij}}{\sum_{i,j=1}^n (\mathbf{K} \Theta)_{ij}}$$

Therefore, $\forall i \in \mathbb{N}_n$, we have:

$$v_i = \sum_{j=1}^n (\mathbf{K}^{-1})_{ij} - \sum_{j=1}^c \mathbf{M}_{ij} - \psi \sum_{j=1}^n \Theta_{ij}$$

From Remark 4.4, we can rewrite (4.30) as:

$$\alpha_{ij} = \mathbf{M}_{ij} + \frac{\zeta_j \sum_{p=1}^n \Theta_{ip}}{\sum_{p,q=1}^n (\mathbf{K} \Theta)_{pq}} + \frac{1}{c} v_i \quad (4.45)$$

From (4.45), we obtain Corollary 4.15, which describes an efficient method to compute the inductive solution of CLapRLS.

Corollary 4.15 (Inductive classification using CLapRLS). *From (4.45), the output class of $\mathbf{x} \in \mathbb{R}^d$ is given by:*

$$\begin{aligned} y(\mathbf{x}) &= \arg \max_{j \in \mathbb{N}_c} \sum_{i=1}^n \mathcal{H}(\mathbf{x}, \mathbf{x}_i) \alpha_{ij} \\ &= \arg \max_{j \in \mathbb{N}_c} \sum_{i=1}^n \mathcal{H}(\mathbf{x}, \mathbf{x}_i) \left(\mathbf{M}_{ij} + \frac{\zeta_j \sum_{p=1}^n \Theta_{ip}}{\sum_{p,q=1}^n (\mathbf{K} \Theta)_{pq}} \right) \end{aligned} \quad (4.46)$$

Moreover, if we also assume that $\omega = \mathbf{1}_c/c$, then:

$$y(\mathbf{x}) = \arg \max_{j \in \mathbb{N}_c} \sum_{i=1}^n \mathcal{H}(\mathbf{x}, \mathbf{x}_i) \left(\mathbf{M}_{ij} - \frac{(\sum_{z=1}^n (\mathbf{KM})_{zj}) \sum_{p=1}^n \Theta_{ip}}{\sum_{p,q=1}^n (\mathbf{K}\Theta)_{pq}} \right) \quad (4.47)$$

Proof. Since v_i in (4.45) does not depend on class j , we get (4.46). Moreover, if we also assume that $\omega = \mathbf{1}_c/c$, we obtain:

$$y(\mathbf{x}) = \arg \max_{j \in \mathbb{N}_c} \sum_{i=1}^n \mathcal{H}(\mathbf{x}, \mathbf{x}_i) \left(\mathbf{M}_{ij} + \frac{(\frac{n}{c} - \sum_{z=1}^n (\mathbf{KM})_{zj}) \sum_{p=1}^n \Theta_{ip}}{\sum_{p,q=1}^n (\mathbf{K}\Theta)_{pq}} \right)$$

This is equivalent to (4.47). $\square$

From Corollary 4.15, we get the output class of $\mathbf{x}$ using CLapRLS as:

$$y(\mathbf{x}) = \arg \max_{j \in \mathbb{N}_c} \sum_{i=1}^n \mathcal{H}(\mathbf{x}, \mathbf{x}_i) \bar{\alpha}_{ij} \quad (4.48)$$

such that

$$\bar{\alpha}_{ij} = \begin{cases} \mathbf{M}_{ij} - \frac{(\sum_{z=1}^n (\mathbf{KM})_{zj}) \sum_{p=1}^n \Theta_{ip}}{\sum_{p,q=1}^n (\mathbf{K}\Theta)_{pq}}, & \text{if } \omega = \mathbf{1}_c/c \\ \mathbf{M}_{ij} + \frac{\zeta_j \sum_{p=1}^n \Theta_{ip}}{\sum_{p,q=1}^n (\mathbf{K}\Theta)_{pq}}, & \text{otherwise} \end{cases} \quad (4.49)$$

Alg. 4 summarizes our theoretical analysis for CLapRLS for induction.

Algorithm 4 – CLapRLS for induction

Input: a graph Laplacian $\mathbf{L} \in \mathbb{R}^{n \times n}$, a nonnegative diagonal matrix $\Sigma \in \mathbb{R}^{n \times n}$, a label matrix $\mathbf{Y} \in \mathbb{B}^{n \times c}$, and regularization parameters $\gamma_A, \gamma_l \in \mathbb{R}_+^*$

1. Compute $\Theta = (\Sigma \mathbf{K} + \gamma_A \mathbf{I}_n + \gamma_l \mathbf{L} \mathbf{K})^{-1}$
2. Compute $\mathbf{M} = \Theta \Sigma \mathbf{Y}$
3. Compute $\zeta \in \mathbb{R}^{1 \times c}$ as in (4.44)
4. Compute $\bar{\alpha} \in \mathbb{R}^{n \times c}$ as in (4.49)
5. Compute $y(\mathbf{x})$ as in (4.48)

Output: the classification function $y(\mathbf{x})$

4.4 Generalization to any distribution

Eq. (2.20) was formulated assuming that $p(\mathbf{x}_i) \approx 1/n$. Such an approximation usually does not reflect the data distribution, possibly leading to poor results for some parameter settings. In this section, we generalize CGFHF, CLGC, and CLapRLS for a general distribution, providing a more flexible regularization framework.

Consider $\mathbf{R} := \begin{bmatrix} \mathbf{R}_{\mathcal{L}} & \mathbf{O}_{l \times u} \\ \mathbf{O}_{u \times l} & \mathbf{R}_{\mathcal{U}} \end{bmatrix} \in \mathbb{R}^{n \times n}$ a diagonal matrix such that $\mathbf{R}_{ii}$ is a measure associated to $\mathbf{x}_i$. Clearly, $\mathbf{R}_{\mathcal{L}} \in \mathbb{R}^{l \times l}$ and $\mathbf{R}_{\mathcal{U}} \in \mathbb{R}^{u \times u}$ are also diagonal matrices. Assuming, $\forall i \in \mathbb{N}_n$,

$$p(\mathbf{x}_i) = \frac{\mathbf{R}_{ii}}{\sum_{j=1}^n \mathbf{R}_{ij}} = \frac{\mathbf{R}_{ii}}{\mathbf{1}_n^\top \mathbf{R} \mathbf{1}_n}$$

we can rewrite (2.20) as:

$$p(\mathbb{C}_k) = \sum_{i=1}^n p(\mathbf{x}_i) p(\mathbb{C}_k | \mathbf{x}_i) = \frac{1}{\mathbf{1}_n^\top \mathbf{R} \mathbf{1}_n} \sum_{i=1}^n \mathbf{R}_{ii} \mathbf{F}_{ik} = \frac{1}{\mathbf{1}_n^\top \mathbf{R} \mathbf{1}_n} (\mathbf{R} \mathbf{F})_{\cdot, k} \quad (4.50)$$

From (4.50), we obtain the following constraint:

$$\mathbf{F}^\top \mathbf{R} \mathbf{1}_n = \mathbf{1}_n^\top \mathbf{R} \mathbf{1}_n \boldsymbol{\omega} \quad (4.51)$$

4.4.1 Generalized CGFHF

Applying $\mathbf{R}$ (or the constraint in (4.51)) in (4.1) yields the following constrained optimization problem:

$$\begin{aligned} & \min_{\mathbf{F} \in \mathbb{R}^{n \times c}} \text{tr}(\mathbf{F}^\top \mathbf{L} \mathbf{F}) \\ \text{s.t. } & \mathbf{F}_{\mathcal{L}} = \mathbf{Y}_{\mathcal{L}}, \mathbf{F} \mathbf{1}_c = \mathbf{1}_n, \mathbf{F}^\top \mathbf{R} \mathbf{1}_n = \mathbf{1}_n^\top \mathbf{R} \mathbf{1}_n \boldsymbol{\omega} \end{aligned} \quad (4.52)$$

If $\mathbf{F}_{\mathcal{L}} = \mathbf{Y}_{\mathcal{L}}$, (4.51) yields:

$$\begin{bmatrix} \mathbf{Y}_{\mathcal{L}}^\top & \mathbf{F}_{\mathcal{U}}^\top \end{bmatrix} \begin{bmatrix} \mathbf{R}_{\mathcal{L}} & \mathbf{O}_{l \times u} \\ \mathbf{O}_{u \times l} & \mathbf{R}_{\mathcal{U}} \end{bmatrix} \begin{bmatrix} \mathbf{1}_l \\ \mathbf{1}_u \end{bmatrix} = \mathbf{Y}_{\mathcal{L}}^\top \mathbf{R}_{\mathcal{L}} \mathbf{1}_l + \mathbf{F}_{\mathcal{U}}^\top \mathbf{R}_{\mathcal{U}} \mathbf{1}_u = \mathbf{1}_n^\top \mathbf{R} \mathbf{1}_n \boldsymbol{\omega}$$

Therefore:

$$\mathbf{F}_{\mathcal{U}}^\top \mathbf{R}_{\mathcal{U}} \mathbf{1}_u = \mathbf{1}_n^\top \mathbf{R} \mathbf{1}_n \boldsymbol{\omega} - \mathbf{Y}_{\mathcal{L}}^\top \mathbf{R}_{\mathcal{L}} \mathbf{1}_l \quad (4.53)$$

As in (2.4), if $\mathbf{F}_{\mathcal{L}} = \mathbf{Y}_{\mathcal{L}}$, we have:

$$\text{tr}(\mathbf{F}^\top \mathbf{L} \mathbf{F}) = \text{tr}(\mathbf{Y}_{\mathcal{L}}^\top \mathbf{L}_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}}) + \text{tr}(\mathbf{F}_{\mathcal{U}}^\top \mathbf{L}_{\mathcal{U}} \mathbf{F}_{\mathcal{U}}) + 2 \text{tr}(\mathbf{F}_{\mathcal{U}}^\top \mathbf{L}_{\mathcal{U}} \mathbf{Y}_{\mathcal{L}})$$

From (4.53) and (2.4), we can rewrite (4.52) as:

$$\begin{aligned} \min_{\mathbf{F}_{\mathcal{U}} \in \mathbb{R}^{u \times c}} \quad & \text{tr} \left(\mathbf{F}_{\mathcal{U}}^{\top} \mathbf{L}_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}} \right) + 2\text{tr} \left(\mathbf{F}_{\mathcal{U}}^{\top} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \right) \\ \text{s.t.} \quad & \mathbf{F}_{\mathcal{U}} \mathbf{1}_c = \mathbf{1}_u, \quad \mathbf{F}_{\mathcal{U}}^{\top} \mathbf{R}_{\mathcal{U}} \mathbf{1}_u = \mathbf{1}_n^{\top} \mathbf{R} \mathbf{1}_n \omega - \mathbf{Y}_{\mathcal{L}}^{\top} \mathbf{R}_{\mathcal{L}} \mathbf{1}_l \end{aligned} \quad (4.54)$$

The minimization problem in (4.54) is the generalized version of CGFHF. Proposition 4.16 finds the closed-form solution of (4.54). Corollary 4.17 shows the solution of Generalized CGFHF for $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$. Corollary 4.18 shows that the constraint $\mathbf{F}_{\mathcal{U}} \mathbf{1}_c = \mathbf{1}_u$ is always satisfied in (4.54) for $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$. Therefore, this constraint can be dropped from the optimization problem without changing the closed-form solution.

Proposition 4.16 (Solution of Generalized CGFHF). *The closed-form solution of (4.54) is given by:*

$$\mathbf{F}_{\mathcal{U}} = -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^{\top} \mathbf{R}_{\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \zeta + \frac{1}{c} \mathbf{v} \mathbf{1}_c^{\top} \quad (4.55)$$

in which

$$\begin{aligned} \zeta &= \mathbf{1}_n^{\top} \mathbf{R} \mathbf{1}_n \omega^{\top} - \mathbf{1}_l^{\top} \mathbf{R}_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \mathbf{1}_u^{\top} \mathbf{R}_{\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ \mathbf{v} &= \mathbf{1}_u + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l - \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^{\top} \mathbf{R}_{\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \mathbf{v}^{(0)} \\ \mathbf{v}^{(0)} &= \mathbf{1}_u^{\top} \mathbf{R}_{\mathcal{U}} \mathbf{1}_u + \mathbf{1}_u^{\top} \mathbf{R}_{\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l \end{aligned} \quad (4.56)$$

Proof. Follow the same steps shown in Proposition 4.1. □

Corollary 4.17 (Generalized CGFHF for $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$). *If $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$, (4.55) is reduced to:*

$$\mathbf{F}_{\mathcal{U}} = -(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^{\top} \mathbf{R}_{\mathcal{U}} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \zeta \quad (4.57)$$

in which

$$\zeta = \mathbf{1}_n^{\top} \mathbf{R} \mathbf{1}_n \omega^{\top} - \mathbf{1}_l^{\top} \mathbf{R}_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \mathbf{1}_u^{\top} \mathbf{R}_{\mathcal{U}} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \quad (4.58)$$

Proof. Follow the same steps shown in Corollary 4.2. □

Corollary 4.18. *If $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$, the solution of (4.54) is equivalent to the solution of the following optimization problem:*

$$\begin{aligned} \min_{\mathbf{F}_{\mathcal{U}} \in \mathbb{R}^{u \times c}} \quad & \text{tr} \left(\mathbf{F}_{\mathcal{U}}^{\top} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}} \right) + 2\text{tr} \left(\mathbf{F}_{\mathcal{U}}^{\top} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \right) \\ \text{s.t.} \quad & \mathbf{F}_{\mathcal{U}}^{\top} \mathbf{R}_{\mathcal{U}} \mathbf{1}_u = \mathbf{1}_n^{\top} \mathbf{R} \mathbf{1}_n \omega - \mathbf{Y}_{\mathcal{L}}^{\top} \mathbf{R}_{\mathcal{L}} \mathbf{1}_l \end{aligned}$$

Proof. The proof comes directly from Corollary 4.2 and the steps shown in Proposition 4.1. $\square$

4.4.2 Generalized CLGC

Applying $\mathbf{R}$ (or the constraint in (4.51)) in (4.12) yields the following constrained optimization problem:

$$\begin{aligned} \min_{\mathbf{F} \in \mathbb{R}^{n \times c}} \quad & \text{tr} \left(\mathbf{F}^\top \mathbf{L} \mathbf{F} + (\mathbf{F} - \mathbf{Y})^\top \Sigma (\mathbf{F} - \mathbf{Y}) \right) \\ \text{s.t.} \quad & \mathbf{F} \mathbf{1}_c = \mathbf{1}_n, \quad \mathbf{F}^\top \mathbf{R} \mathbf{1}_n = \mathbf{1}_n^\top \mathbf{R} \mathbf{1}_n \omega \end{aligned} \quad (4.59)$$

The minimization problem in (4.59) is the generalized version of CLGC. Proposition 4.19 finds the closed-form solution of (4.59). Corollary 4.20 shows that Generalized CGFHF is a special case of Generalized CLGC. Corollary 4.21 shows that Generalized RMGT is a special case of Generalized CLGC.

Proposition 4.19 (*Solution of Generalized CLGC*). *The closed-form solution of (4.59) is given by:*

$$\mathbf{F} = \Theta \Sigma \mathbf{Y} + \frac{\Theta \mathbf{1}_n}{\mathbf{1}_n^\top \mathbf{R} \Theta \mathbf{1}_n} \zeta + \frac{1}{c} \mathbf{v} \mathbf{1}_c^\top \quad (4.60)$$

in which

$$\begin{aligned} \Theta &= (\mathbf{L} + \Sigma)^{-1} \\ \zeta &= \mathbf{1}_n^\top \mathbf{R} \mathbf{1}_n \omega^\top - \mathbf{1}_n^\top \mathbf{R} \Theta \Sigma \mathbf{Y} \\ \mathbf{v} &= \mathbf{1}_n - \Theta \Sigma \mathbf{Y} \mathbf{1}_c - \frac{\Theta \mathbf{1}_n}{\mathbf{1}_n^\top \mathbf{R} \Theta \mathbf{1}_n} \mathbf{v}^{(0)} \\ \mathbf{v}^{(0)} &= \mathbf{1}_n^\top \mathbf{R} \mathbf{1}_n - \mathbf{1}_n^\top \mathbf{R} \Theta \Sigma \mathbf{Y} \mathbf{1}_c \end{aligned} \quad (4.61)$$

Proof. Follow the same steps shown in Proposition 4.6. $\square$

Corollary 4.20 (*Generalized CGFHF as special case of Generalized CLGC*). *If $\Sigma_{\mathcal{U}} = \mathbf{O}_{u \times u}$ and $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$, (4.60) is reduced to:*

$$\mathbf{F} \approx \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{R}_{\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \zeta + \frac{1}{c} \mathbf{v} \mathbf{1}_c^\top \end{bmatrix}$$

in which

$$\begin{aligned}
\zeta &= \mathbf{1}_n^\top \mathbf{R} \mathbf{1}_n \omega^\top - \mathbf{1}_l^\top \mathbf{R}_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \mathbf{1}_u^\top \mathbf{R}_{\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\
\mathbf{v} &= \mathbf{1}_u + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l - \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{R}_{\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \mathbf{v}^{(0)} \\
\mathbf{v}^{(0)} &= \mathbf{1}_u^\top \mathbf{R}_{\mathcal{U}} \mathbf{1}_u + \mathbf{1}_u^\top \mathbf{R}_{\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l
\end{aligned}$$

Proof. Follow the same steps shown in Proposition 4.7. $\square$

Corollary 4.21 (Generalized RMGT as special case of Generalized CLGC). Consider $\Sigma_{\mathcal{U}} = \mathbf{O}_{u \times u}$ and $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$. If we also have $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$, (4.60) is reduced to (4.57), approximately.

Proof. Follow the same steps shown in Corollary 4.8. $\square$

4.4.3 Generalized CLapRLS

Applying $\mathbf{R}$ (or the constraint in (4.51)) in (4.29) yields the following constrained optimization problem:

$$\begin{aligned}
\min_{\alpha \in \mathbb{R}^{n \times c}} \quad & \text{tr} \left((\mathbf{K}\alpha - \mathbf{Y})^\top \Sigma (\mathbf{K}\alpha - \mathbf{Y}) + \gamma_A \alpha^\top \mathbf{K} \alpha + \gamma_I \alpha^\top \mathbf{K} \mathbf{L} \mathbf{K} \alpha \right) \\
\text{s.t.} \quad & \mathbf{K} \alpha \mathbf{1}_c = \mathbf{1}_n, \quad \alpha^\top \mathbf{K} \mathbf{R} \mathbf{1}_n = \mathbf{1}_n^\top \mathbf{R} \mathbf{1}_n \omega
\end{aligned} \tag{4.62}$$

The minimization problem in (4.62) is the generalized version of CLapRLS. Proposition 4.22 finds the closed-form solution of Generalized CLapRLS. Corollaries 4.23, 4.24, and 4.25 show the inductive forms of Generalized CLGC, Generalized CGFHF, and Generalized RMGT, respectively. These corollaries show that the corresponding methods are naturally inductive and special cases of Generalized CLapRLS.

Proposition 4.22 (Solution of Generalized CLapRLS). The closed-form solution of (4.62) is given by:

$$\alpha = \Theta \Sigma \mathbf{Y} + \frac{\Theta \mathbf{1}_n}{\mathbf{1}_n^\top \mathbf{R} \mathbf{K} \Theta \mathbf{1}_n} \zeta + \frac{1}{c} \mathbf{v} \mathbf{1}_c^\top \tag{4.63}$$

in which

$$\begin{aligned}
\Theta &= (\Sigma \mathbf{K} + \gamma_A \mathbf{I}_n + \gamma_I \mathbf{L} \mathbf{K})^{-1} \\
\zeta &= \mathbf{1}_n^\top \mathbf{R} \mathbf{1}_n \omega^\top - \mathbf{1}_n^\top \mathbf{R} \mathbf{K} \Theta \Sigma \mathbf{Y} \\
\mathbf{v} &= \mathbf{K}^{-1} \mathbf{1}_n - \Theta \Sigma \mathbf{Y} \mathbf{1}_c - \frac{\Theta \mathbf{1}_n}{\mathbf{1}_n^\top \mathbf{R} \mathbf{K} \Theta \mathbf{1}_n} \mathbf{v}^{(0)} \\
\mathbf{v}^{(0)} &= \mathbf{1}_n^\top \mathbf{R} \mathbf{1}_n - \mathbf{1}_n^\top \mathbf{R} \mathbf{K} \Theta \Sigma \mathbf{Y} \mathbf{1}_c
\end{aligned} \tag{4.64}$$

Therefore, the transductive solution is given by:

$$\mathbf{F} = \mathbf{K}\Theta\Sigma\mathbf{Y} + \frac{\mathbf{K}\Theta\mathbf{1}_n}{\mathbf{1}_n^\top \mathbf{R}\mathbf{K}\Theta\mathbf{1}_n} \zeta + \frac{1}{c} \mathbf{K}\mathbf{v}\mathbf{1}_c^\top \quad (4.65)$$

Proof. Follow the same steps shown in Proposition 4.11. $\square$

Corollary 4.23 (Inductive form of Generalized CLGC). If $\gamma_A = 0$ and $\gamma_I = 1$, (4.63) is reduced to:

$$\alpha = \mathbf{K}^{-1} \left(\Theta\Sigma\mathbf{Y} + \frac{\Theta\mathbf{1}_n}{\mathbf{1}_n^\top \mathbf{R}\Theta\mathbf{1}_n} \zeta + \frac{1}{c} \mathbf{v}\mathbf{1}_c^\top \right) \quad (4.66)$$

in which

$$\begin{aligned} \Theta &= (\mathbf{L} + \Sigma)^{-1} \\ \zeta &= \mathbf{1}_n^\top \mathbf{R}\mathbf{1}_n \omega^\top - \mathbf{1}_n^\top \mathbf{R}\Theta\Sigma\mathbf{Y} \\ \mathbf{v} &= \mathbf{1}_n - \Theta\Sigma\mathbf{Y}\mathbf{1}_c - \frac{\Theta\mathbf{1}_n}{\mathbf{1}_n^\top \mathbf{R}\Theta\mathbf{1}_n} \mathbf{v}^{(0)} \\ \mathbf{v}^{(0)} &= \mathbf{1}_n^\top \mathbf{R}\mathbf{1}_n - \mathbf{1}_n^\top \mathbf{R}\Theta\Sigma\mathbf{Y}\mathbf{1}_c \end{aligned}$$

Therefore, the transductive solution is given by:

$$\mathbf{F} = \left(\Theta\Sigma\mathbf{Y} + \frac{\Theta\mathbf{1}_n}{\mathbf{1}_n^\top \mathbf{R}\Theta\mathbf{1}_n} \zeta + \frac{1}{c} \mathbf{v}\mathbf{1}_c^\top \right)$$

Proof. Substitute $\gamma_A = 0$ and $\gamma_I = 1$ in (4.63) to obtain (4.66). $\square$

Corollary 4.24 (Inductive form of Generalized CGFHF). If $\Sigma_{\mathcal{U}} = \mathbf{O}_{u \times u}$ and $\Sigma_{\mathcal{L}}^{-1} \rightarrow \mathbf{O}_{l \times l}$, (4.60) is reduced to:

$$\alpha \approx \mathbf{K}^{-1} \left[\begin{array}{c} \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{R}_{\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \zeta + \frac{1}{c} \mathbf{v}\mathbf{1}_c^\top \end{array} \right] \quad (4.67)$$

in which

$$\begin{aligned} \zeta &= \mathbf{1}_n^\top \mathbf{R}\mathbf{1}_n \omega^\top - \mathbf{1}_l^\top \mathbf{R}_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \mathbf{1}_u^\top \mathbf{R}_{\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ \mathbf{v} &= \mathbf{1}_u + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l - \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{R}_{\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \mathbf{v}^{(0)} \\ \mathbf{v}^{(0)} &= \mathbf{1}_u^\top \mathbf{R}_{\mathcal{U}} \mathbf{1}_u + \mathbf{1}_u^\top \mathbf{R}_{\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l \end{aligned}$$

Therefore, the transductive solution is given by:

$$\mathbf{F} \approx \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{R}_{\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \boldsymbol{\zeta} + \frac{1}{c} \mathbf{v} \mathbf{1}_c^\top \end{bmatrix}$$

Proof. Applying Proposition 4.7 in (4.66) yields (4.67). $\square$

Corollary 4.25 (Inductive form of Generalized RMGT). *If $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$, (4.67) is reduced to:*

$$\boldsymbol{\alpha} \approx \mathbf{K}^{-1} \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -(\mathbf{L}_{\mathbb{C}})^{-1}_{\mathcal{U}\mathcal{U}} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{(\mathbf{L}_{\mathbb{C}})^{-1}_{\mathcal{U}\mathcal{U}} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{R}_{\mathcal{U}} (\mathbf{L}_{\mathbb{C}})^{-1}_{\mathcal{U}\mathcal{U}} \mathbf{1}_u} \boldsymbol{\zeta} \end{bmatrix} \quad (4.68)$$

in which

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

Therefore, the transductive solution is given by:

$$\mathbf{F} \approx \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -(\mathbf{L}_{\mathbb{C}})^{-1}_{\mathcal{U}\mathcal{U}} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{(\mathbf{L}_{\mathbb{C}})^{-1}_{\mathcal{U}\mathcal{U}} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{R}_{\mathcal{U}} (\mathbf{L}_{\mathbb{C}})^{-1}_{\mathcal{U}\mathcal{U}} \mathbf{1}_u} \boldsymbol{\zeta} \end{bmatrix}$$

Proof. The proof comes directly from Corollary 4.2. $\square$

4.5 Our findings

In this chapter, we provided a theoretical study on constrained graph-based SSL. Our findings in this chapter can be summarized as follows:

1. we provided three new graph-based SSL algorithms applying the normalization constraints of Liu and Chang (2009) into the regularization framework of existing methods. We called the proposed constrained methods CGFHF, CLGC, and CLapRLS;
2. we showed that CGFHF, CLGC, and CLapRLS have closed-form solution, which is computed with the same time complexity of widely used graph-based SSL algorithms (ZHU; GHAHRAMANI; LAFFERTY, 2003; ZHOU *et al.*, 2004; BELKIN; NIYOGI; SINDHWANI, 2006; LIU; CHANG, 2009);
3. we proved that CGFHF generalizes RMGT to *any* graph Laplacian, which adds flexibility to the regularization framework;
4. we proved that CLGC generalizes CGFHF and RMGT, incorporating a penalty term in the objective function with regularization parameters for both labeled and unlabeled examples;

5. we proved that CLapRLS generalizes CLGC, CGFHF, and RMGT. This means that the proposed methods are naturally inductive;
6. we showed that the kernel function has no influence in the transductive solution of CGFHF and CLGC;
7. we proposed optimized ways to compute CGFHF, CLGC, and CLapRLS under some conditions;
8. we generalized one of the constraints of [Liu and Chang \(2009\)](#) to *any* distribution. From this new constraint, we generalized CGFHF, CLGC, and CLapRLS to any distribution. We called the corresponding methods Generalized CGFHF, Generalized CLGC, and Generalized CLapRLS.

EXPERIMENTAL SETUP

In this chapter, we describe the experimental design decisions that we have taken to facilitate the reproduction of our results. Most of our experimental setup comes from the study of [Sousa, Rezende and Batista \(2013\)](#).

This chapter is organized as follows. Section [5.1](#) describes the two sets of experiments we performed. Section [5.2](#) describes the graph construction methods we used with their corresponding parameter setting. Section [5.3](#) shows how we used the graph Laplacians in our experiments. Section [5.4](#) describes the parameter settings for the SSL algorithms. Section [5.5](#) describes the empirical evaluation models of [Sousa, Rezende and Batista \(2013\)](#), used to analyze our results. Finally, Section [5.6](#) provides some concluding remarks.

5.1 Sets of experiments

In this thesis, we provide the following two sets of experiments:

- experiments on time series transductive classification;
- experiments on constrained graph-based SSL.

For the first experiment, we evaluate GFHF, LGC, LapRLS, and RMGT on time series data. We used 45 data sets from the UCR repository¹, which are described in Table [1](#).

For the second experiment, we evaluate the effectiveness of the proposed constrained methods against GFHF, LGC, LapRLS, LapSVM, and RMGT. More precisely, we used the USPS, COIL₂, DIGIT-1, G-241C, G-241N, and TEXT data sets, which are freely available² ([CHAPELLE; SCHÖLKOPF; ZIEN, 2006](#)). These data sets are described in Table [2](#). The

¹ http://www.cs.ucr.edu/~eamonn/time_series_data/.

² <http://olivier.chapelle.cc/ssl-book/benchmarks.html>.

Table 1 – Description of the time series data sets used in the first experiment.

Data sets	# labeled (%)	# unlabeled (%)	series length	# classes	% minority	% majority	ID
50words	450 (49.72)	455 (50.28)	270	50	0.22	11.56	D1
Adiac	390 (49.93)	391 (50.07)	176	37	1.03	3.85	D2
Beef	30 (50.0)	30 (50.0)	470	5	20.0	20.0	D3
CBF	30 (3.26)	900 (96.77)	128	3	26.67	40.0	D4
ChlorineConcentration	467 (10.84)	3840 (89.16)	166	3	19.49	56.10	D5
Cinc-ECG-torso	40 (2.81)	1380 (97.19)	639	4	12.5	32.5	D6
Coffee	28 (50.0)	28 (50.0)	286	2	50.0	50.0	D7
Cricket-X	390 (50.0)	390 (50.0)	300	12	6.41	10.0	D8
Cricket-Y	390 (50.0)	390 (50.0)	300	12	7.18	9.23	D9
Cricket-Z	390 (50.0)	390 (50.0)	300	12	6.41	10.51	D10
DiatomSizeReduction	16 (4.97)	306 (95.03)	345	4	6.25	37.50	D11
ECG200	100 (50.0)	100 (50.0)	96	2	31.0	69.0	D12
ECGFiveDays	23 (2.60)	861 (97.40)	136	2	39.13	60.87	D13
FaceAll	560 (24.89)	1690 (75.11)	131	14	7.14	7.14	D14
FaceFour	24 (21.43)	88 (78.57)	350	4	12.50	33.33	D15
FacesUCR	200 (8.88)	2050 (91.12)	132	14	2.0	16.50	D16
Fish	175 (50.0)	175 (50.0)	463	7	12.0	16.0	D17
Gun-Point	50 (25.0)	150 (75.0)	150	2	48	52	D18
Haptics	155 (33.47)	308 (66.53)	1092	5	11.61	23.23	D19
InlineSkate	100 (15.38)	550 (84.62)	1882	7	9.0	18.0	D20
ItalyPowerDemand	67 (6.11)	1029 (93.89)	24	2	49.25	50.75	D21
Lightning2	60 (49.59)	61 (50.41)	637	2	33.33	66.67	D22
Lightning7	70 (48.95)	73 (51.05)	319	7	7.14	27.14	D23
Mallat	55 (2.29)	2345 (97.71)	1024	8	3.64	20.0	D24
MedicalImages	381 (33.39)	760 (66.61)	99	10	1.58	53.28	D25
MoteStrain	20 (1.57)	1252 (98.43)	84	2	50.0	50.0	D26
NonInvasiveFetalECG-Thorax1	1800 (47.81)	1965 (52.19)	750	42	1.94	3.11	D27
NonInvasiveFetalECG-Thorax2	1800 (47.81)	1965 (52.19)	750	42	1.94	3.11	D28
OliveOil	30 (50.0)	30 (50.0)	570	4	13.33	43.33	D29
OSULeaf	200 (45.24)	242 (54.76)	427	6	7.5	26.5	D30
SonyAIBORobotSurface	20 (3.22)	601 (96.78)	70	2	30.0	70.0	D31
SonyAIBORobotSurfaceII	27 (2.76)	953 (97.24)	65	2	40.74	59.26	D32
StarLightCurves	1000 (10.82)	8236 (89.18)	1024	3	15.2	57.3	D33
SwedishLeaf	500 (44.44)	625 (55.56)	128	15	5.20	8.40	D34
Symbols	25 (2.45)	995 (97.55)	398	6	12.0	32.0	D35
synthetic-control	300 (50.0)	300 (50.0)	60	6	16.66	16.66	D36
Trace	100 (50.0)	100 (50.0)	275	4	21.0	31.0	D37
TwoLeadECG	23 (1.98)	1139 (98.02)	82	2	47.83	52.17	D38
Two-Patterns	1000 (20.0)	4000 (80.0)	128	4	23.70	27.10	D39
uWaveGestureLibrary-X	896 (20.01)	3582 (79.99)	315	8	11.16	14.17	D40
uWaveGestureLibrary-Y	896 (20.01)	3582 (79.99)	315	8	11.16	14.17	D41
uWaveGestureLibrary-Z	896 (20.01)	3582 (79.99)	315	8	11.16	14.17	D42
Wafer	1000 (13.95)	6164 (86.05)	152	2	9.7	90.3	D43
WordsSynonyms	267 (29.50)	638 (70.50)	270	25	0.75	22.47	D44
yoga	300 (9.09)	3000 (90.91)	426	2	45.67	54.33	D45

document containing extensive experimental results for GFHF, LGC, LapRLS, LapSVM, and RMGT in these six data sets is freely available³.

In high-dimensional data, the distance to the nearest neighbor approaches the distance of the farthest neighbor (BEYER *et al.*, 1999). It degenerates the quality of the graph and possibly decreases the performance of SSL algorithms. Therefore, we used the pre-processing step suggested by Sousa, Rezende and Batista (2013).

Specifically, we ran *principal component analysis* (PCA) to reduce the dimensionality of the data sets to 50 using the *Matlab Toolbox for Dimensionality Reduction*⁴ library. We did not run PCA only on the TEXT data set to maintain the sparseness property of these data. The

³ <https://www.researchgate.net/publication/316634962_Empirical_evaluation_models_for_graph-based_semi-supervised_learning_supplementary_material>.

⁴ <http://homepage.tudelft.nl/19j49/Matlab_Toolbox_for_Dimensionality_Reduction.html>.

pre-processed data sets are freely available⁵.

Table 2 – Description of the data sets used in the second experiment.

Data sets	# examples (n)	# features (d)	# classes (c)	% minority	% majority
COIL ₂	1500	241	2	50.0	50.0
DIGIT-1	1500	241	2	48.93	51.07
G-241C	1500	241	2	50.0	50.0
G-241N	1500	241	2	49.87	50.13
TEXT	1500	11960	2	50.0	50.0
USPS	1500	241	2	20.0	80.0

5.2 Graph construction

This section describes the algorithms and parameter settings chosen for graph construction.

5.2.1 Distance function

Given a training sample $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n \subset \mathbb{R}^d$, we generated a distance matrix $\Psi \in \mathbb{R}^{n \times n}$ such that $\Psi_{ij} := \Psi(\mathbf{x}_i, \mathbf{x}_j)$, $\forall i, j \in \mathbb{N}_n$, where $\Psi(\cdot, \cdot)$ is a distance function defined in the input space. For the first set of experiments, we used the *Dynamic Time Warping* (DTW)⁶ as distance function for all data sets. For the second set of experiments, we used the *cosine* distance for the TEXT data set and the L_2 -norm for the other data sets. Given $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$, the L_2 -norm is defined by:

$$\|\mathbf{u} - \mathbf{v}\|_2 = \sqrt{\sum_{i=1}^n (\mathbf{u}_i - \mathbf{v}_i)^2}$$

Assuming that $\langle \cdot, \cdot \rangle_n$ is the inner product of vectors in $\mathbb{R}^n$, the cosine distance is defined by:

$$\cos(\mathbf{u}, \mathbf{v}) = 1 - \frac{\langle \mathbf{u}, \mathbf{v} \rangle_n}{\|\mathbf{u}\|_2 \|\mathbf{v}\|_2}$$

5.2.2 Adjacency graph construction

From Ψ , we generated the adjacency matrix $\hat{\mathbf{A}} \in \mathbb{B}^{n \times n}$ using the *k-nearest neighbors* (kNN) graph. Therefore, we set $\hat{\mathbf{A}}_{ij} = 1$ if and only if $\mathbf{x}_j$ is one of the k closest examples of $\mathbf{x}_i$. Mathematically, generating the kNN graph can be viewed as solving the following optimization problem (JEBARA; WANG; CHANG, 2009):

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

⁶ DTW is an algorithm for measuring similarity between two time series which may vary in length. This algorithm calculates an optimal match between two given time series with certain restrictions. A good review on DTW can be found in (KEOGH, 2002; MUEEN; KEOGH, 2016).

$$\begin{aligned} \min_{\hat{\mathbf{A}} \in \mathbb{B}^{n \times n}} \quad & \text{tr}(\hat{\mathbf{A}}^\top \Psi) = \sum_{i=1}^n \sum_{j=1}^n \hat{\mathbf{A}}_{ij} \Psi(\mathbf{x}_i, \mathbf{x}_j) \\ \text{s.t.} \quad & \hat{\mathbf{A}} \mathbf{1}_n = k \mathbf{1}_n, \quad \text{tr}(\hat{\mathbf{A}}) = 0 \end{aligned}$$

Since the adjacency matrix of a k NN graph may not be symmetric, three strategies are commonly used to symmetrize it:

- *mutual kNN* (mutKNN), which generates $\mathbf{A} = \min(\hat{\mathbf{A}}, \hat{\mathbf{A}}^\top)$. This means that $\mathbf{A}_{ij} = 1$ just if $\hat{\mathbf{A}}_{ij} = \hat{\mathbf{A}}_{ji} = 1$. We have $\mathbf{A}_{ij} = 0$ in the other cases;
- *symmetric kNN* (symKNN), which generates $\mathbf{A} = \max(\hat{\mathbf{A}}, \hat{\mathbf{A}}^\top)$. This means that $\mathbf{A}_{ij} = 0$ just if $\hat{\mathbf{A}}_{ij} = \hat{\mathbf{A}}_{ji} = 0$. We have $\mathbf{A}_{ij} = 1$ in the other cases;
- *symmetry-favored kNN* (symFKNN) (LIU; CHANG, 2009), which generates $\mathbf{A} = \hat{\mathbf{A}} + \hat{\mathbf{A}}^\top$ (a non-binary adjacency matrix).

Therefore, we used symKNN, mutKNN, and symFKNN as adjacency graph construction methods in our experiments. Since the mutKNN graph may contain isolated vertices for small values of k , we created an undirected edge between each isolated vertex and its nearest neighbor, as suggested by Sousa, Rezende and Batista (2013). Other strategies may also be applied as well (OZAKI *et al.*, 2011). The parameter k was chosen in the set $\{4, 6, 8, \dots, 40\}$.

5.2.3 Weighted matrix generation

From $\mathbf{A}$ and $\Psi(\cdot, \cdot)$ (or also Ψ), we have to generate a sparse, nonnegative, symmetric, weighted matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$ where $\mathbf{W}_{ij}$ encodes similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$. In order to generate $\mathbf{W}$, we used the following weighted matrix generation methods: RBF kernel; similarity function of Hein & Maier's (HM) (HEIN; MAIER, 2007); and *Locally Linear Embedding* (LLE) (ROWEIS; SAUL, 2000). The RBF kernel is defined by:

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

where $\sigma \in \mathbb{R}_+^*$ is the kernel's parameter. When there is a very restricted number of labeled examples, it is not straightforward to find an adequate value for σ . We estimated the value of σ by:

$$\sigma = \frac{1}{3n} \sum_{i=1}^n \Psi\left(\mathbf{x}_i, \mathbf{x}_i^{(k)}\right) \quad (5.1)$$

as suggested by Jebara, Wang and Chang (2009). The HM method is defined by:

$$\mathcal{K}_k(\mathbf{x}_i, \mathbf{x}_j) = \exp \left(- \frac{\Psi^2(\mathbf{x}_i, \mathbf{x}_j)}{\left(\max \left\{ \Psi(\mathbf{x}_i, \mathbf{x}_i^{(k)}), \Psi(\mathbf{x}_j, \mathbf{x}_j^{(k)}) \right\} \right)^2} \right)$$

As discussed by [Sousa, Rezende and Batista \(2013\)](#), the HM method can be viewed as an RBF kernel with a local adaptive kernel size σ_{ij}^k for each ordered pair of neighbors $(\mathbf{x}_i, \mathbf{x}_j)$. For the HM method, we compute the σ_{ij}^k 's as follows:

$$\sigma_{ij}^k = \frac{\sqrt{2}}{2} \max \left\{ \Psi(\mathbf{x}_i, \mathbf{x}_i^{(k)}), \Psi(\mathbf{x}_j, \mathbf{x}_j^{(k)}) \right\}$$

The RBF kernel and the HM method generate $\mathbf{W}$ as $\mathbf{W}_{ij} = \mathbf{A}_{ij} \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$, $\forall i, j \in \mathbb{N}_n$. On the other hand, the LLE method generates $\mathbf{W}$ by solving the following optimization problem:

$$\begin{aligned} \min_{\mathbf{W} \in \mathbb{R}^{n \times n}} \sum_{i=1}^n \left\| \mathbf{x}_i - \sum_{\mathbf{x}_j \in \mathcal{N}_i} \mathbf{W}_{ij} \mathbf{x}_j \right\|_2^2 \\ \text{s.t. } \mathbf{W} \mathbf{1}_n = \mathbf{1}_n, \mathbf{W} \geq 0 \end{aligned} \quad (5.2)$$

In order to efficiently solve (5.2), we used the *Local Anchor Embedding* (LAE) method ([LIU; HE; CHANG, 2010](#)). As discussed by [Sousa, Rezende and Batista \(2013\)](#), LLE can be viewed as a special case of LAE when we use a bipartite graph whose “*anchor*” (or *landmark*) points are exactly the training examples. Since LLE may not generate a symmetric weighted matrix, we symmetrize the output matrix of LLE, $\mathbf{W}_{LLE}$, as follows:

$$\mathbf{W} = \frac{1}{2} (\mathbf{W}_{LLE} + \mathbf{W}_{LLE}^\top)$$

This process is used for symKNN-LLE and mutKNN-LLE. Because the adjacency matrix of the symFKNN graph is non-binary, we compute $\widehat{\mathbf{W}} = \mathbf{W}_{LLE} \odot \mathbf{A}$ where $\odot$ is the *Hadamard*⁷ (or element-wise) product. Then, we generate the weighted matrix for symFKNN-LLE as follows:

$$\mathbf{W} = \frac{1}{2} (\widehat{\mathbf{W}} + \widehat{\mathbf{W}}^\top)$$

Since a graph construction method is a combination of an adjacency graph construction method with a weighted matrix generation method, we use in our experiments the following graph construction methods: symKNN-RBF (G1); mutKNN-RBF (G2); symFKNN-RBF (G3); symKNN-HM (G4); mutKNN-HM (G5); symFKNN-HM (G6); symKNN-LLE (G7); mutKNN-LLE (G8); and symFKNN-LLE (G9).

For simplicity, the expressions “*the symKNN-RBF method*” and “*the symKNN-RBF graph construction method*” are equivalent to “*the symKNN-RBF graph*”. We refer to graphs generated

⁷ Given two matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{a \times b}$, we compute $\mathbf{C} = \mathbf{A} \odot \mathbf{B}$ as $\mathbf{C}_{ij} = \mathbf{A}_{ij} \mathbf{B}_{ij}$, $\forall i \in \mathbb{N}_a, \forall j \in \mathbb{N}_b$.

by the RBF kernel (symKNN-RBF, mutKNN-RBF, and symFKNN-RBF) as RBF-based graphs. This notation is also applied to the other graph construction methods.

5.3 Graph Laplacian generation

Since the normalized Laplacian $\mathbf{L}_\mathbb{N}$ achieves better results than the combinatorial Laplacian $\mathbf{L}_\mathbb{C}$ in general SSL tasks (JOHNSON; ZHANG, 2007), we used $\mathbf{L}_\mathbb{N}$ in the graph-based SSL algorithms when applicable. We obtained poor results using $\mathbf{L}_\mathbb{N}$ in RMGT during preliminary experiments. Therefore, we used the combinatorial Laplacian for RMGT and the normalized Laplacian for the other SSL algorithms, as suggested by Sousa, Rezende and Batista (2013).

In preliminary experiments, we observed some numerical instabilities using RMGT in the COIL₂ data set. These instabilities occurred because at least one of the eigenvalues of the graph Laplacian was equal to (or approximately) zero. In an attempt to avoid numerical instabilities while solving linear systems using the graph Laplacians, we generated $\mathbf{L}_\mathbb{C}$ and $\mathbf{L}_\mathbb{N}$ as follows (SOUSA; REZENDE; BATISTA, 2013):

$$\begin{aligned}\mathbf{L}_\mathbb{C} &= \eta \mathbf{D} - \mathbf{W} \\ \mathbf{L}_\mathbb{N} &= \eta \mathbf{I}_n - \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2}\end{aligned}$$

where a small $\eta > 1$ is used to increase the eigenvalues of the graph Laplacians. In our experiments, we set $\eta = 1.01$.

Although this simple procedure performed well, avoiding numerical instabilities on most data sets, we observed numerical instabilities while running RMGT on some time series data. Even increasing the value of η , these numerical instabilities were observed. We provide comments on this issue in Chapter 6.

5.4 Graph-based SSL algorithms

Given a graph Laplacian $\mathbf{L}$, we are able to perform classification using graph-based SSL algorithms. As suggested by Sousa, Rezende and Batista (2013), the values of μ in LGC and CLGC were chosen in the set $\{0.01, 0.05, 0.1, 0.5, 1, 2, 5, 10, 50, 10^2\}$. As suggested by Melacci and Belkin (2011), the values of γ_A and γ_I in LapRLS, LapSVM, and CLapRLS were chosen in the set $\{10^{-6}, 10^{-4}, 10^{-2}, 10^{-1}, 1, 10, 10^2\}$. We generated the kernel matrix $\mathbf{K}$ for LapRLS, LapSVM, and CLapRLS using the RBF kernel with the same parameter estimation in (5.1), as suggested by Sousa, Rezende and Batista (2013).

We ran LapSVM using the source code in (MELACCI; BELKIN, 2011)⁸. We trained the LapSVMs using Newton's method, which gave better results than the preconditioned conjugate

⁸ <<http://www.dii.unisi.it/~melacci/lapsvmp/index.html>>.

gradient method during preliminary experiments. For RMGT, CGFHF, CLGC, and CLapRLS, we assumed a uniform class distribution ($\omega = \mathbf{1}_c/c$), as suggested by [Liu and Chang \(2009\)](#).

In order to classify the unlabeled examples, we used the *class mass normalization* (CMN) procedure ([ZHU; GHAHRAMANI; LAFFERTY, 2003](#)). The multi-class version of CMN can be viewed as follows:

$$y(\mathbf{x}_i) = \arg \max_{j \in \mathbb{N}_c} \frac{\omega_j \mathbf{F}_{ij}}{\mathbf{1}_n^\top \mathbf{F} \cdot j}$$

This is a useful procedure when we are dealing with unbalanced data sets. We obtained poor results using CMN in RMGT, CGFHF, CLGC, and CLapRLS in preliminary experiments. Therefore, we report the results for these four algorithms using the *argmax* operator. This yields the classification function $y(\mathbf{x}_i) = \arg \max_{j \in \mathbb{N}_c} \mathbf{F}_{ij}$ for CLGC and CLapRLS and $y(\mathbf{x}_i) = \arg \max_{j \in \mathbb{N}_c} (\mathbf{F}_{\mathcal{U}})_{i-l,j}$ for RMGT and CGFHF.

We report the results for GFHF, LGC, and LapRLS using CMN while the results for LapSVM are reported using the *sign* function. This yields the following classification function:

$$y(\mathbf{x}) = \begin{cases} \text{positive class} & , f(\mathbf{x}) \geq 0 \\ \text{negative class} & , f(\mathbf{x}) < 0 \end{cases}$$

For GFHF, we computed CMN using $\mathbf{F}_{\mathcal{U}}$ instead of $\mathbf{F}$, as suggested by [Zhu, Ghahramani and Lafferty \(2003\)](#).

5.5 Empirical evaluation models

[Sousa, Rezende and Batista \(2013\)](#) proposed the following three empirical evaluation models for graph-based SSL: (1) *evaluation of classifier stability*; (2) *evaluation of graph stability*; and (3) *evaluation of error surfaces*⁹. Fig. 1 presents illustrative examples for these evaluation models.

5.5.1 Best case analysis

In the best case analysis, we compare each combination of SSL algorithm and graph construction method using their lowest average error rates on each data set over all parameter values. We also used two ranking-based approaches for this analysis. In the first approach, we focus on answering the following question:

⁹ The original name in ([SOUSA; REZENDE; BATISTA, 2013](#)) is *evaluation of regularization parameters*. We can easily extend such an evaluation for any pair of parameters. Therefore, we can analyze the error surfaces for an SSL algorithm corresponding to this pair of parameters.

Figure 1 – Illustrative examples for the following empirical evaluation models: (a) *evaluation of classifier stability*; (b) *evaluation of graph stability*; and (c) *evaluation of error surfaces*.

What are the best SSL algorithms for a given graph construction method in terms of average rankings over all data sets?

In order to answer this question, we initially compute the average rankings of the SSL algorithms for each graph construction method over all data sets. Let $r_{i,j}$ be the average ranking of the SSL algorithm i for the graph construction method j . Then, we compare the averages $\bar{r}_i = \frac{1}{a} \sum_{j=1}^a r_{i,j}$, where a is the number of graph construction methods.

In the second approach, we focus on answering the following question:

What are the best graph construction methods for a given SSL algorithm in terms of average rankings over all data sets?

In order to answer this question, we provide a similar computation to the first approach. Specifically, we compare the averages $\bar{s}_i = \frac{1}{b} \sum_{j=1}^b s_{i,j}$, where b is the number of SSL algorithms and $s_{i,j}$ is the average ranking of the graph construction method i for the SSL algorithm j .

5.5.2 Evaluation of classifier stability

In the evaluation of classifier stability, we evaluate the SSL algorithms' performance with respect to k for each graph construction method. More precisely, we want to identify *why* and *when* an SSL algorithm outperforms another for a given parameter setting and when such an algorithm achieves stable performance with respect to k , if applicable. Moreover, we are interested in finding parameter settings in which a given SSL algorithm fails and provide insights for future research on how to overcome possible weaknesses.

In practice, an SSL algorithm may achieve the best overall performance only for a very narrow range of the parameter values and a mediocre performance for most parameter values. Then, the evaluation of classifier stability is an invaluable tool to identify which SSL

algorithms provide a good trade-off between performance and stability with respect to k (SOUSA; REZENDE; BATISTA, 2013). In this evaluation, we focus on answering the following question:

Given a partially labeled training sample, how can we evaluate an SSL algorithm's performance with respect to k for a predefined graph construction method?

When the SSL algorithm has no regularization parameter, the experimental analysis is straightforward because the performance is just a function of one variable. Analyzing the results over many variables may be hard due to high performance variation and high dimensionality of the function space. Our approach to overcome this problem is to look for the best performance and keep the corresponding values for the parameters, excluding k . Then, we analyze the SSL algorithm's performance with respect to k for these parameters.

5.5.3 Evaluation of graph stability

In the evaluation of graph stability, we evaluate the performance of the graph construction methods with respect to k for a given SSL algorithm. Specifically, we want to verify how the graph construction methods affect the performance of a given SSL algorithm. In this evaluation, we focus on answering the following question:

Given a partially labeled training sample, how can we evaluate the impact of graph construction on the performance of a predefined SSL algorithm with respect to k ?

When the SSL algorithm has no regularization parameters, we use a similar approach to that adopted in the previous evaluation. However, when the SSL algorithm has regularization parameters, this problem can have many possible solutions. As in the previous evaluation, our approach is to look for the best performance and keep the corresponding values for the parameters, excluding k . Then, we analyze the performance of an SSL algorithm with respect to k over many graph construction methods for these parameters.

5.5.4 Evaluation of error surfaces

In the evaluation of error surfaces, we evaluate the performance of an SSL algorithm with respect to two parameters for a given graph construction method. In this evaluation, we focus on answering the following question:

Given two parameters, how can we evaluate an SSL algorithm's performance with respect to these parameters for a predefined graph construction method?

Our approach to answer this question is equivalent to those for the previous evaluations. Consider $\{\vartheta_i\}_{i=1}^a$ a set of a parameters. Without loss of generality, suppose we want to analyze

the performance of an SSL algorithm with respect to ϑ_1 and ϑ_2 for a given graph construction method. Therefore, we look for the best performance and keep the corresponding value for the parameters, excluding ϑ_1 and ϑ_2 . This means we fix the values of $\{\vartheta_i\}_{i=3}^a$. Then, we analyze the SSL algorithm's performance with respect to ϑ_1 and ϑ_2 for these fixed parameter values.

5.6 Concluding remarks

We described in this chapter the experimental settings and the empirical evaluation models used in our experiments. In Chapters 6 and 7, we will show how to analyze the experimental results of graph-based SSL using the empirical evaluation models of [Sousa, Rezende and Batista \(2013\)](#).

TIME SERIES TRANSDUCTIVE CLASSIFICATION ON GRAPHS

In this chapter, we provide our experimental analysis on time series transductive classification on graphs. We used the experimental setup described in Chapter 5 and the empirical evaluation models described in Section 5.5.

This chapter contains the results in (SOUSA; SOUZA; BATISTA, 2014; SOUSA; SOUZA; BATISTA, 2015) with additional experimental results. The document containing extensive experimental results on time series transductive classification is freely available¹. The data sets and source codes used in these experiments are also freely available².

We organized this chapter as follows. Section 6.1 analyzes the results for the best case analysis. Section 6.2 analyzes the results for the evaluation of classifier stability. Section 6.3 analyzes the results for the evaluation of graph stability. Section 6.4 analyzes the results for the evaluation of error surfaces. Section 6.5 provides a discussion on some experimental issues and how they can be overcome. Finally, Section 6.6 describes our findings for time series transductive classification.

6.1 Best case analysis

Tables 3, 4, and 5 show the best error rates for the SSL algorithms combined with the graph construction methods over all parameter values for the data sets described in Table 1. We also report the results for the 1NN classifier using the following distance measures:

- *Euclidean Distance* (ED);

¹ <https://www.researchgate.net/publication/316649218_Time_series_transductive_classification_on_graphs_supplementary_material>.

² <http://sites.labc.icmc.usp.br/sousa/time_series/>.

Table 3 – Error rates (%) for the SSL algorithms for each graph construction method on the D1–D15 data sets.

Data sets	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15
GFHF-symKNN-RBF	19.78	37.34	46.67	0.0	7.00	0.07	14.29	20.26	16.92	14.87	0.65	8.0	4.18	2.43	7.95
GFHF-mutKNN-RBF	19.34	36.57	46.67	0.0	6.22	0.0	10.71	20.77	17.18	16.41	0.0	9.0	3.72	17.93	6.82
GFHF-symFKNN-RBF	19.56	36.06	46.67	0.0	6.59	0.07	14.29	20.0	16.67	15.38	0.65	8.0	4.53	2.43	7.95
GFHF-symKNN-HM	23.30	41.18	43.33	0.0	13.83	0.43	14.29	22.82	17.44	17.18	1.96	8.0	6.27	4.79	5.68
GFHF-mutKNN-HM	21.98	36.83	43.33	0.0	11.41	0.0	10.71	21.79	16.92	18.20	0.0	13.0	3.83	18.82	6.82
GFHF-symFKNN-RBF	22.20	37.60	46.67	0.0	13.20	0.29	17.86	21.03	15.90	15.64	1.96	8.0	6.27	3.90	5.68
GFHF-symKNN-LLE	26.37	33.25	33.33	0.0	0.73	0.65	7.14	26.67	20.26	22.31	1.96	8.0	2.21	4.56	11.36
GFHF-mutKNN-LLE	23.74	33.76	23.33	0.0	0.52	0.0	3.57	24.10	19.74	18.97	0.0	8.0	3.72	18.40	9.09
GFHF-symFKNN-LLE	25.27	31.97	33.33	0.0	0.70	0.07	7.14	25.38	17.95	20.51	1.96	8.0	2.09	4.32	10.23
LGC-symKNN-RBF	18.90	39.90	43.33	0.0	7.5	0.0	14.29	19.74	15.90	15.38	89.22	10.0	4.30	2.19	5.68
LGC-mutKNN-RBF	18.90	35.29	43.33	0.0	6.67	0.0	10.71	20.0	17.18	16.41	89.22	9.0	3.72	16.69	6.82
LGC-symFKNN-RBF	18.90	37.08	43.33	0.0	7.00	0.0	14.29	19.49	16.67	15.38	89.22	9.0	4.30	2.01	6.82
LGC-symKNN-HM	22.64	39.39	43.33	0.0	14.11	0.14	7.14	22.82	17.69	17.44	1.96	10.0	6.04	11.72	5.68
LGC-mutKNN-HM	20.0	35.29	43.33	0.0	11.41	0.0	10.71	21.03	17.44	17.95	0.0	12.0	3.83	17.87	6.82
LGC-symFKNN-HM	21.10	37.85	43.33	0.0	13.44	0.14	10.71	20.51	17.44	16.41	1.96	9.0	5.92	8.70	5.68
LGC-symKNN-LLE	25.27	32.74	33.33	0.0	1.07	0.72	3.57	26.15	20.26	19.23	1.96	9.0	2.44	5.92	10.23
LGC-mutKNN-LLE	21.54	33.25	26.67	0.0	0.55	0.0	3.57	22.05	18.72	17.95	0.0	10.0	4.07	17.28	7.95
LGC-symFKNN-LLE	23.74	31.97	33.33	0.0	0.88	0.43	3.57	25.13	18.20	18.97	1.96	9.0	2.32	5.27	10.23
LapRLS-symKNN-RBF	18.90	36.06	30.0	0.0	7.03	0.0	3.57	20.51	15.64	14.87	0.65	8.0	4.06	2.19	4.54
LapRLS-mutKNN-RBF	18.24	33.76	30.0	0.0	6.25	0.0	3.57	19.49	15.64	15.64	0.0	9.0	3.48	8.70	5.68
LapRLS-symFKNN-RBF	18.90	34.53	30.0	0.0	6.64	0.0	3.57	20.0	16.15	15.13	0.65	8.0	4.30	2.01	4.54
LapRLS-symKNN-HM	20.66	37.08	30.0	0.0	13.75	0.14	0.0	21.79	16.41	16.15	1.96	8.0	5.92	4.32	5.68
LapRLS-mutKNN-HM	20.0	34.53	30.0	0.0	11.30	0.0	3.57	20.77	16.15	15.90	0.0	8.0	3.48	11.24	6.82
LapRLS-symFKNN-HM	20.22	36.83	30.0	0.0	13.05	0.14	0.0	20.51	16.41	15.64	1.96	7.0	5.92	4.02	5.68
LapRLS-symKNN-LLE	21.10	28.64	30.0	0.0	0.73	0.72	3.57	22.31	17.44	16.15	1.96	8.0	1.63	4.02	6.82
LapRLS-mutKNN-LLE	20.22	30.43	26.67	0.0	0.52	0.0	3.57	21.79	17.69	15.64	0.0	8.0	3.37	11.24	6.82
LapRLS-symFKNN-LLE	20.88	27.88	30.0	0.0	0.70	0.07	3.57	22.31	16.92	16.15	1.96	7.0	1.39	3.67	6.82
RMGT-symKNN-RBF	20.22	46.04 [†]	46.67	0.0	7.00	0.14	10.71	22.05	19.23	15.64	‡	11.0	4.18	16.69	3.41
RMGT-mutKNN-RBF	19.34	38.87 [†]	46.67	0.0	6.35	0.07	10.71	21.54	18.97	16.41	‡	9.0	4.30	18.40	6.82
RMGT-symFKNN-RBF	19.56	42.97 [†]	46.67	0.0	6.69	0.14	10.71	21.28	18.97	15.64	‡	11.0	4.30	16.57	4.54
RMGT-symKNN-HM	21.32	37.08	50.0	0.0	18.75	0.07	14.29	21.28	17.95	16.67	1.96	8.0	6.50	9.64	5.68
RMGT-mutKNN-HM	20.88	34.01	46.67	0.0	15.99	0.0	17.86	20.51	18.72	17.18	0.0	6.0	7.08	17.69	6.82
RMGT-symFKNN-HM	19.34	36.32	46.67	0.0	17.66	0.07	17.86	20.26	16.92	16.15	1.96	7.0	6.85	16.63	4.54
RMGT-symKNN-LLE	25.05	31.46	33.33	0.0	6.07	0.0	7.14	26.67	19.23	20.51	1.31	9.0	3.72	16.39	7.95
RMGT-mutKNN-LLE	22.64	31.97	30.0	0.0	5.70	0.0	3.57	22.56	19.49	19.23	0.0	8.0	5.92	17.22	7.95
RMGT-symFKNN-LLE	24.83	28.90	36.67	0.0	5.81	0.0	7.14	24.62	17.95	18.72	1.31	9.0	4.65	16.51	5.68
ED	36.90	38.90	46.70	14.80	35.0	10.30	25.0	42.60	35.60	38.0	6.50	12.0	20.30	28.60	21.60
DTW	24.20	39.10	46.70	0.4	35.0	7.0	17.90	23.60	19.70	18.0	6.50	12.0	20.30	19.20	11.40
CID	22.60	37.90	46.70	0.10	35.10	5.40	17.90	24.90	19.70	20.50	6.50	–	21.80	14.40	12.50
TSBF	20.90	24.50	28.70	0.9	33.60	26.20	0.40	27.80	25.90	26.30	12.60	–	18.30	23.40	5.10
PROP	18.0	35.30	36.70	0.20	36.0	6.20	0.0	20.30	15.60	15.60	5.90	–	17.80	15.20	9.10
ST	–	48.60	16.70	0.10	33.30	13.70	0.0	–	–	–	12.70	–	0.0	25.40	10.20
RPCD	22.64	38.36	36.67	–	48.91	2.10	0.0	29.23	26.15	29.23	3.59	14.0	13.59	19.05	5.68
TSF	27.70	26.10	30.0	3.90	26.0	6.90	7.10	28.70	20.0	23.90	10.10	–	7.0	23.10	3.40
BoP	46.60	43.20	43.30	1.30	3.60	–	–	–	–	–	–	–	–	21.90	2.30
LS	–	41.40	43.30	11.40	38.20	30.10	3.60	–	–	–	19.90	–	0.60	34.10	51.10
FS	–	51.40	44.70	5.30	41.70	17.40	6.80	–	–	–	11.70	–	0.40	41.10	9.0
TE	35.20	35.80	40.0	17.10	–	–	21.40	–	–	–	–	–	–	28.10	14.80

- *Dynamic Time Warping* (DTW);
- *Complexity-Invariant Distance* (CID) (BATISTA *et al.*, 2014);
- *Time Series Bag of Features* (TSBF) (BAYDOGAN; RUNGER; TUV, 2013);
- *Proportional Elastic Ensemble* (PROP) (LINES; BAGNALL, 2014);
- *Shapelet Transform* (ST) (HILLS *et al.*, 2013);
- *Recurrence Patterns Compression Distance* (RPCD) (SILVA; SOUZA; BATISTA, 2013);
- *Time Series Forests* (TSF) (DENG *et al.*, 2013);
- *Bag of Patterns* (BoP) (LIN; KHADE; LI, 2012);

Table 4 – Error rates (%) for the SSL algorithms for each graph construction method on the D16–D30 data sets.

Data sets	D16	D17	D18	D19	D20	D21	D22	D23	D24	D25	D26	D27	D28	D29	D30
GFHF-symKNN-RBF	3.61	13.14	6.0	55.19	59.27	2.62	13.11	19.18	5.07	20.39	10.70	13.49	10.08	13.33	35.95
GFHF-mutKNN-RBF	4.58	13.71	5.33	54.87	56.54	2.43	11.47	19.18	5.07	20.39	14.22	12.67	10.13	13.33	35.54
GFHF-symFKNN-RBF	3.56	13.14	5.33	56.82	59.09	2.43	13.11	19.18	4.18	20.0	11.66	12.52	10.18	13.33	35.95
GFHF-symKNN-HM	5.17	15.43	10.67	52.60	59.09	2.62	16.39	17.81	6.27	22.37	8.31	15.32	10.23	10.0	38.84
GFHF-mutKNN-HM	4.93	15.43	7.33	52.60	56.36	2.43	9.84	20.55	5.93	21.97	14.78	13.23	10.38	13.33	37.19
GFHF-symFKNN-HM	4.78	14.86	8.67	52.27	59.09	2.62	14.75	17.81	5.29	21.71	8.63	14.05	9.62	13.33	38.02
GFHF-symKNN-LLE	6.19	10.29	2.0	51.30	59.64	2.92	14.75	17.81	5.33	23.42	12.38	10.58	7.58	10.0	37.60
GFHF-mutKNN-LLE	5.27	10.86	5.33	52.60	56.54	2.43	9.84	19.18	6.61	22.24	15.49	11.30	8.55	13.33	39.67
GFHF-symFKNN-LLE	5.51	9.71	2.67	50.97	59.82	2.82	13.11	19.18	4.39	22.89	12.70	9.97	7.63	10.0	37.60
LGC-symKNN-RBF	3.85	86.29	5.33	55.52	57.64	2.53	13.11	17.81	4.26	19.21	12.06	13.03	10.58	10.0	36.78
LGC-mutKNN-RBF	4.58	86.29	6.0	55.19	55.09	2.53	8.20	19.18	4.26	19.60	15.02	12.72	10.28	13.33	35.12
LGC-symFKNN-RBF	3.85	86.29	5.33	55.52	57.27	2.53	11.47	19.18	4.01	19.21	12.38	12.87	10.53	10.0	35.95
LGC-symKNN-HM	5.41	13.71	9.33	52.27	57.27	2.91	11.47	17.81	4.82	21.58	8.23	15.42	10.18	10.0	38.84
LGC-mutKNN-HM	5.32	14.29	6.0	51.62	55.09	2.43	8.20	19.18	4.39	21.18	15.33	13.28	10.48	13.33	36.78
LGC-symFKNN-HM	4.78	13.14	10.0	51.62	56.54	2.91	9.84	17.81	4.35	21.32	8.07	13.64	9.87	10.0	37.60
LGC-symKNN-LLE	6.19	10.86	3.33	50.97	58.91	3.01	13.11	20.55	4.56	23.42	12.06	10.53	7.84	10.0	37.19
LGC-mutKNN-LLE	4.93	10.29	6.0	52.27	56.54	2.43	8.20	21.92	4.69	21.97	15.89	11.50	8.80	13.33	36.36
LGC-symFKNN-LLE	5.37	9.14	3.33	50.97	58.18	2.82	13.11	21.92	4.09	22.63	12.38	9.82	7.48	10.0	36.36
LapRLS-symKNN-RBF	3.61	12.0	4.0	54.54	58.36	2.53	11.47	17.81	4.60	20.26	10.46	12.67	9.82	10.0	35.54
LapRLS-mutKNN-RBF	3.95	12.57	5.33	54.54	58.36	2.43	6.56	19.18	2.86	20.0	11.26	12.47	9.87	13.33	35.12
LapRLS-symFKNN-RBF	3.46	13.14	5.33	54.54	58.54	2.53	9.84	17.81	3.92	20.13	10.38	12.57	9.87	13.33	35.54
LapRLS-symKNN-HM	5.12	13.71	6.0	52.60	57.64	2.72	11.47	17.81	5.07	22.10	8.15	13.03	9.87	13.33	35.95
LapRLS-mutKNN-HM	4.19	12.57	6.0	52.60	56.36	2.43	8.20	19.18	2.39	20.39	12.06	12.47	9.92	13.33	35.95
LapRLS-symFKNN-HM	4.78	14.29	6.0	51.95	57.45	2.62	11.47	17.81	4.73	21.05	8.07	12.93	9.62	13.33	35.54
LapRLS-symKNN-LLE	5.90	11.43	2.0	50.97	57.09	2.91	11.47	19.18	4.90	22.10	10.46	10.43	7.63	10.0	35.95
LapRLS-mutKNN-LLE	4.49	10.86	4.67	51.30	56.54	2.33	8.20	19.18	2.77	20.79	12.06	11.20	8.35	13.33	35.95
LapRLS-symFKNN-LLE	5.37	10.29	2.67	50.0	57.82	2.72	11.47	17.81	4.22	21.97	10.38	9.77	7.63	13.33	35.95
RMGT-symKNN-RBF	4.78	‡	6.67	58.77	60.54	3.21	24.59	19.18	9.34	22.63	10.70	16.44	‡	6.67	40.50
RMGT-mutKNN-RBF	4.44	‡	6.0	57.47	60.0	3.01	13.11	23.29	7.21	21.58	13.98	13.69	‡	10.0	38.43
RMGT-symFKNN-RBF	4.58	‡	7.33	58.12	60.73	3.11	22.95	20.55	8.53	21.97	10.94	14.96	‡	6.67	39.26
RMGT-symKNN-HM	4.98	15.43	10.67	53.25	55.64	2.91	16.39	20.55	5.33	31.18	7.67	15.01	10.18	6.67	38.84
RMGT-mutKNN-HM	4.88	14.29	6.0	52.60	56.36	2.43	19.67	26.03	3.45	29.34	13.34	12.82	9.92	10.0	37.60
RMGT-symFKNN-HM	4.44	14.29	6.0	52.27	55.64	2.82	13.11	20.55	3.84	31.18	7.75	14.25	9.72	10.0	37.60
RMGT-symKNN-LLE	5.46	9.71	2.0	52.27	58.18	3.01	16.39	21.92	3.03	33.82	12.06	10.08	7.18	13.33	35.95
RMGT-mutKNN-LLE	4.49	10.86	4.67	51.95	55.64	2.43	14.75	24.66	3.41	31.05	14.62	10.94	7.99	13.33	35.54
RMGT-symFKNN-LLE	4.88	9.71	2.67	51.30	58.18	3.01	16.39	24.66	2.22	32.24	11.82	9.36	7.38	16.67	35.95
ED	23.10	21.70	8.70	63.0	65.80	4.50	24.60	42.50	8.60	31.60	12.10	17.10	12.0	13.30	48.30
DTW	8.80	16.0	8.70	58.80	61.30	4.50	13.10	28.80	8.60	25.30	13.40	18.50	12.90	16.70	38.40
CID	10.20	15.40	7.30	57.10	58.60	4.40	13.10	26.0	7.50	25.80	20.50	–	–	16.70	37.20
TSBF	9.0	8.0	1.10	48.80	60.30	9.60	25.70	26.20	3.70	26.90	13.50	13.80	13.0	9.0	32.90
PROP	6.30	3.40	0.70	58.40	56.70	3.90	11.50	23.30	5.0	24.50	11.40	17.80	11.20	13.30	19.40
ST	–	2.90	1.30	–	–	5.20	39.30	23.30	7.40	39.30	11.10	–	–	–	–
RPCD	5.85	12.57	0.0	61.36	68.0	15.74	24.59	35.62	–	28.95	20.29	–	–	16.67	35.54
TSF	10.90	15.40	4.70	56.50	67.50	3.30	18.0	26.30	7.20	23.20	11.80	10.30	9.40	10.0	42.60
BoP	–	7.40	9.30	–	–	–	16.40	46.60	–	–	–	–	–	–	–
LS	33.80	22.30	10.70	–	–	6.40	57.40	45.20	34.40	41.30	16.80	–	–	–	–
FS	32.80	19.70	6.10	–	–	9.50	29.50	40.30	3.30	43.30	21.70	–	–	–	–
TE	–	19.40	5.30	–	–	–	23.0	30.10	–	–	–	–	–	–	–

- *Logical Shapelets* (LS) (YE; KEOGH, 2009);
- *Fast Shapelets* (FS) (RAKTHANMANON; KEOGH, 2013);
- *Transformation based Ensembles* (TE) (BAGNALL *et al.*, 2012).

The results for ED and DTW can be found in the UCR repository’s site³. The results for the other distance measures can be found in (LINES; BAGNALL, 2014).

In Tables 3, 4, and 5, the symbol † indicates that the corresponding SSL algorithm combined with a given graph construction method found no solution for *some* parameter values while the symbol ‡ indicates that no solution was found for *any* parameter value, including the

³ <http://www.cs.ucr.edu/~eamonn/time_series_data/>.

Table 5 – Error rates (%) for the SSL algorithms for each graph construction method on the D31–D45 data sets.

Data sets	D31	D32	D33	D34	D35	D36	D37	D38	D39	D40	D41	D42	D43	D44	D45
GFHF-symKNN-RBF	2.66	8.71	6.85	8.48	2.11	0.33	0.0	0.09	0.0	19.96	26.49	27.33	0.26	23.35	11.83
GFHF-mutKNN-RBF	0.50	5.04	5.43	7.84	1.51	0.0	1.0	0.09	0.0	18.37	25.71	25.91	0.32	24.45	11.17
GFHF-symFKNN-RBF	2.16	8.50	6.33	7.2	2.11	0.33	0.0	0.09	0.0	19.51	26.19	27.08	0.26	23.51	11.77
GFHF-symKNN-HM	23.29	14.17	6.28	10.72	3.01	1.0	2.0	0.09	0.0	20.60	27.02	28.36	0.28	25.86	12.43
GFHF-mutKNN-HM	0.83	4.93	5.10	8.0	1.61	0.0	3.0	0.09	0.02	18.12	26.30	25.26	0.41	25.70	13.47
GFHF-symFKNN-HM	14.14	14.06	5.80	8.64	2.81	0.67	2.0	0.09	0.0	19.65	26.63	27.72	0.23	25.86	11.90
GFHF-symKNN-LLE	21.80	14.27	4.89	7.2	2.31	0.67	5.0	0.0	0.05	21.36	28.03	28.64	0.36	29.62	12.33
GFHF-mutKNN-LLE	1.16	5.77	5.04	8.16	1.61	0.0	3.0	0.09	0.05	18.84	26.60	25.57	0.52	28.21	13.37
GFHF-symFKNN-LLE	11.81	14.17	4.39	6.56	2.31	0.33	5.0	0.0	0.05	20.41	27.25	27.39	0.29	27.90	11.93
LGC-symKNN-RBF	2.0	8.18	6.22	9.44	2.41	0.0	0.0	0.09	0.0	19.57	26.44	26.66	0.26	22.26	11.93
LGC-mutKNN-RBF	0.50	5.35	5.23	8.96	1.51	0.0	0.0	0.09	0.0	18.84	25.71	26.47	0.32	23.51	11.43
LGC-symFKNN-RBF	2.16	7.87	5.80	8.96	2.11	0.0	0.0	0.09	0.0	19.26	26.30	26.47	0.26	22.73	11.70
LGC-symKNN-HM	7.65	11.54	5.65	10.88	3.12	0.33	2.0	0.09	0.0	20.88	27.81	28.92	0.28	26.02	12.80
LGC-mutKNN-HM	0.83	4.93	5.16	9.12	1.51	0.0	1.0	0.09	0.0	18.51	26.55	25.46	0.39	25.23	13.43
LGC-symFKNN-HM	5.99	11.44	5.07	9.28	3.01	0.33	1.0	0.09	0.0	19.93	26.97	27.50	0.28	25.23	12.40
LGC-symKNN-LLE	7.15	11.65	4.39	7.36	2.61	0.33	4.0	0.0	0.0	21.61	28.34	29.15	0.31	28.37	12.70
LGC-mutKNN-LLE	1.0	5.98	5.01	8.48	1.51	0.0	1.0	0.09	0.02	19.23	26.88	25.54	0.44	28.06	13.53
LGC-symFKNN-LLE	6.82	10.91	3.97	7.36	2.31	0.33	3.0	0.0	0.0	21.19	27.83	27.44	0.29	26.96	12.13
LapRLS-symKNN-RBF	2.0	8.08	6.39	9.12	2.11	0.0	0.0	0.09	0.0	19.96	26.69	27.50	0.26	23.20	11.53
LapRLS-mutKNN-RBF	0.33	5.35	5.38	8.64	1.51	0.0	0.0	0.09	0.0	18.56	25.91	25.91	0.11	23.51	11.47
LapRLS-symFKNN-RBF	2.16	8.08	6.31	8.16	2.11	0.0	0.0	0.09	0.0	19.60	26.41	27.05	0.26	23.67	11.63
LapRLS-symKNN-HM	8.32	11.54	6.22	10.08	3.01	0.33	1.0	0.09	0.0	20.80	26.97	29.20	0.19	24.29	12.07
LapRLS-mutKNN-HM	0.83	5.35	5.48	9.28	1.61	0.0	0.0	0.09	0.0	18.62	26.80	25.38	0.19	24.45	12.93
LapRLS-symFKNN-HM	6.82	11.33	6.23	8.32	2.81	0.33	1.0	0.09	0.0	19.65	26.88	28.03	0.19	24.45	11.60
LapRLS-symKNN-LLE	8.32	12.07	6.29	7.68	2.31	0.33	3.0	0.0	0.0	21.66	27.30	29.37	0.39	24.29	12.23
LapRLS-mutKNN-LLE	1.0	6.09	5.09	7.68	1.61	0.0	0.0	0.09	0.02	19.12	27.08	26.21	0.23	24.14	12.83
LapRLS-symFKNN-LLE	7.15	11.86	6.20	6.88	2.31	0.33	3.0	0.0	0.0	20.63	27.30	28.39	0.37	24.45	11.83
RMGT-symKNN-RBF	9.82	9.02	13.93	13.44	3.92	0.67	0.0	0.0	0.0	24.18	28.92	31.83	0.73 [†]	24.61	15.30
RMGT-mutKNN-RBF	1.16	4.41	7.72	9.28	1.51	0.33	0.0	0.0	0.0	20.43	25.96	28.78	0.83 [†]	24.61	12.10
RMGT-symFKNN-RBF	7.99	8.81	13.57	12.16	3.72	0.33	0.0	0.0	0.0	23.79	28.48	31.43	0.78 [†]	24.29	14.50
RMGT-symKNN-HM	1.50	15.84	10.59	10.24	2.91	1.0	2.0	0.09	0.0	20.83	27.39	28.34	27.84	23.51	13.37
RMGT-mutKNN-HM	0.17	5.25	10.48	8.64	1.61	0.0	1.0	0.09	0.0	18.01	25.88	25.40	10.85	23.82	13.70
RMGT-symFKNN-HM	1.16	15.74	11.15	9.44	2.81	0.67	1.0	0.09	0.0	19.90	26.86	27.69	28.88	22.41	12.77
RMGT-symKNN-LLE	1.50	16.26	9.64	5.92	2.11	0.67	7.0	0.09	0.05	21.61	27.81	28.34	31.98	28.68	12.90
RMGT-mutKNN-LLE	0.50	9.76	10.26	7.36	1.61	0.0	3.0	0.09	0.05	18.90	26.80	25.01	30.39	27.59	13.30
RMGT-symFKNN-LLE	1.0	16.58	10.27	6.08	2.01	0.33	5.0	0.0	0.05	20.30	27.27	27.02	33.42	27.74	12.20
ED	30.50	14.10	15.10	21.30	10.0	12.0	24.0	25.30	9.0	26.10	33.80	35.0	0.5	38.20	17.0
DTW	30.50	14.10	9.50	15.70	6.20	1.70	1.0	13.20	0.15	22.70	30.10	32.20	0.5	25.20	15.50
CID	18.50	12.30	6.60	11.70	5.90	2.70	1.0	13.80	0.40	21.10	27.80	29.30	0.60	24.30	15.60
TSBF	17.50	19.60	2.20	7.50	3.40	0.80	2.0	4.60	0.10	16.40	24.90	21.70	0.40	30.20	14.90
PROP	29.30	12.40	7.90	8.50	4.90	1.0	1.0	6.70	0.0	19.90	28.30	29.0	0.30	22.60	12.10
ST	10.50	10.90	–	11.80	11.80	2.0	2.0	–	8.20	–	–	–	0.10	–	19.70
RPCD	20.30	15.74	11.83	9.76	9.55	–	–	12.64	–	40.70	37.88	38.33	0.34	27.59	13.40
TSF	23.50	17.70	3.60	10.90	12.10	2.30	0.0	11.20	5.30	21.30	28.80	26.70	4.70	38.10	15.70
BoP	–	–	–	19.80	–	3.70	0.0	–	12.90	–	–	–	0.30	–	17.0
LS	14.0	15.40	–	18.70	35.70	53.0	0.0	14.40	46.10	–	–	–	0.10	–	26.0
FS	31.40	21.50	–	26.90	6.80	8.10	0.20	9.0	11.30	–	–	–	0.40	–	24.90
TE	–	–	–	15.70	–	8.30	20.0	–	16.50	–	–	–	0.20	–	16.30

sparsifier parameter k and the regularization parameters (γ_A , γ_I , or μ), when applicable. The four worst results obtained by an SSL algorithm over all graph construction methods on each data set have a grey background while the best one is in bold. The best error rate over all SSL algorithms and graph construction methods on each data set is boxed.

The SSL algorithms outperformed the supervised learning algorithms by a large margin for many settings of graph construction method on the *CBF* (D4), *DiatomSizeReduction* (D11), *ECGFiveDays* (D13), *SonyAIBORobotSurface* (D31), *Symbols* (D35), and *TwoLeadECG* (D38) data sets. Therefore, graph-based SSL algorithms can be really effective on data sets that have only a few labeled examples. This is a widely used assumption in the SSL literature (ZHU, 2005; CHAPELLE; SCHÖLKOPF; ZIEN, 2006).

The SSL algorithms also showed exceptional performances for many settings of graph construction method on the *Adiac* (D2), *ChlorineConcentration* (D5), *CinC-ECG-torso* (D6), *FacesUCR* (D16), *Haptics* (D19), *InlineSkate* (D20), *Mallat* (D24), *MedicalImages* (D25), *OSULeaf* (D30), *SonyAIBORobotSurface* (D31), *StarLightCurves* (D33), and *Symbols* (D35) data sets. Such an effectiveness on unbalanced data sets is explained by the use of CMN on the output matrix $\mathbf{F}$ for LGC and LapRLS and on $\mathbf{F}_{\mathcal{U}}$ for GFHF. We observed that RMGT showed poor results in comparison to the other SSL algorithms for some graph construction methods on the *Mallat* (D24), *MedicalImages* (D25), *StarLightCurves* (D33), and *Wafer* (D43) data sets.

RMGT showed competitive performance with the other SSL algorithms in most settings of graph construction method and data set. However, we observed *numerical instabilities* when applying RMGT on some data sets using RBF-based graphs. Specifically, RMGT found no solution for the *DiatomSizeReduction* (D11), *Fish* (D17), and *NonInvasiveFetalECG-Thorax2* (D28) data sets using RBF-based graphs for *all* values of k . Comments on this issue can be found in Section 6.5.6.

RMGT found no solution using RBF-based graphs for many values of k on the *Adiac* (D2) and *Wafer* (D43) data sets. Even increasing the value of η in the combinatorial Laplacian $\mathbf{L}_C = \eta \mathbf{D} - \mathbf{W}$, these instabilities still occur. Probably, a global kernel size σ may not be adequate for these data sets when using RMGT. Comments on this issue can be found in Section 6.5.4.

LGC achieved poor results on the *DiatomSizeReduction* (D11) and *Fish* (D17) data sets using RBF-based graphs for *all* parameter values. Specifically, this classifier showed more than 86% of error rate on these data sets using RBF-based graphs while the other SSL algorithms (excluding RMGT) achieved satisfactory results for *all* graph construction methods. However, the results for LGC using HM- and LLE-based graphs are competitive with the other SSL algorithms for all graph construction methods. Comments on this issue can be found in Section 6.5.3.

LLE-based graphs achieved exceptional results in comparison to HM- and RBF-based graphs on the *ChlorineConcentration* (D5) and *Coffee* (D7) data sets using GFHF and LGC. This superiority is also evidenced for RMGT on the *Coffee* (D7) data set. Moreover, on the *ChlorineConcentration* (D5) data set, LapRLS achieved its best results using LLE-based graphs. However, these graph construction methods achieved the worst results on the *FaceFour* (D15) data set for GFHF and LGC. Comments on the effectiveness of LLE-based graphs on some experimental settings can be found in Section 6.5.2.

In order to statistically evaluate our results for the best case analysis, we used non-parametric statistical tests. Since we compared many algorithms over many data sets with no “control” algorithm, we used the Friedman’s test (FRIEDMAN, 1940) with the Nemenyi’s post test (NEMENYI, 1963). In our evaluation, we used a significance level of 0.05. See (DEMSAR, 2006) for a review on statistical tests for machine learning.

Table 6 – Average rankings for the SSL algorithms for each graph construction method.

	G1	G2	G3	G4	G5	G6	G7	G8	G9	mean
GFHF	2.535	2.407	2.465	2.872	2.953	2.895	2.837	2.919	2.884	2.752
LGC	2.291	2.523	2.244	2.733	2.570	2.616	2.709	2.721	2.698	2.567
LapRLS	1.686	1.686	1.791	1.802	1.965	2.01	1.849	1.872	1.895	1.839
RMGT	3.488	3.384	3.5	2.593	2.512	2.477	2.605	2.488	2.523	2.841

Table 7 – Average rankings for the graph construction methods for each SSL algorithm.

	GFHF	LGC	LapRLS	RMGT	mean
symKNN-RBF	4.628	4.663	4.279	6.314	4.971
mutKNN-RBF	3.779	3.977	3.849	4.814	4.105
symFKNN-RBF	3.965	4.151	4.291	5.756	4.541
symKNN-HM	6.488	6.465	6.674	5.546	6.293
mutKNN-HM	4.884	4.779	4.849	4.465	4.744
symFKNN-HM	5.547	5.349	5.547	4.291	5.183
symKNN-LLE	5.860	5.895	5.907	5.186	5.712
mutKNN-LLE	4.802	4.802	4.395	4.221	4.555
symFKNN-LLE	5.046	4.919	5.209	4.407	4.895

Table 6 shows the average rankings for the SSL algorithms for each graph construction method while Table 7 shows the average rankings for the graph construction methods for each SSL algorithm. In Table 6, the best average ranking over the SSL algorithms for each graph construction method is in bold. In Table 7, the best average ranking over the graph construction methods for each SSL algorithm is in bold. In both tables, the average rankings corresponding to the algorithms that were *statistically* outperformed by the best ranked method have a grey background. We also report the overall average rankings for the SSL algorithms and the graph construction methods in Tables 6 and 7, respectively.

Table 6 shows that LapRLS achieved the best average ranking for *all* graph construction methods as well as the best overall average ranking. Therefore, for the best case analysis, LapRLS is the most effective SSL algorithm in terms of average rankings, *independently* of the graph construction method used. In addition, LapRLS statistically outperformed the competing SSL algorithms for most graph construction methods. Comments on the effectiveness of LapRLS on some experimental settings can be found in Section 6.5.7.

Table 6 also shows that RMGT achieved the worst average rankings for RBF-based graphs while achieving the second best average ranking for HM- and LLE-based graphs. Moreover, RMGT showed the worst overall average ranking. Therefore, in terms of average ranking, RBF-based graphs may not be adequate for RMGT. However, this classifier achieved competitive results to the other SSL algorithms on many data sets using RBF-based graphs.

Table 7 shows that mutKNN-RBF achieved the best average ranking for GFHF, LGC, and LapRLS as well as the best overall average ranking. Therefore, for the best case analysis, mutKNN-RBF is the best suited graph construction method for time series transductive classification. In addition, mutKNN-based graphs achieved better average rankings than symKNN- and symFKNN-based graphs for most weighted matrix generation methods and SSL algorithms. The effectiveness of mutKNN-based graphs is also evidenced in the overall average rankings.

Comments on the effectiveness of mutKNN-based graphs on some experimental settings can be found in Section 6.5.1.

RBF-based graphs achieved competitive results in the overall average rankings. The symKNN-RBF and symFKNN-RBF graphs achieved the second and third best average rankings, respectively, for GFHF, LGC, and LapRLS. However, these graphs achieved the worst average rankings for RMGT, possibly not being the most appropriate graph construction methods for this SSL algorithm in terms of average ranking in the best case analysis.

Figure 2 – Error rates for the SSL algorithms with respect to k on the *Gun-Point* (using symKNN-HM), *50words* (using symKNN-LLE), and *OSULeaf* (using symKNN-HM) data sets.

6.2 Evaluation of classifier stability

Fig. 2 shows the error rates for the SSL algorithms with respect to k on the *Gun-Point*, *50words*, and *OSULeaf* data sets for given graph construction methods. The 1NN classifier showed similar results to LapRLS and these two methods outperformed GFHF, LGC, and RMGT by a large margin for $k \geq 10$ for *all* data sets. GFHF, LGC, and RMGT showed similar performances for the entire range of k values for *all* these data sets. LapRLS achieved *exceptional* performance and stability with respect to k . Comments on the effectiveness of LapRLS on some experimental settings can be found in Section 6.5.7.

Fig. 3 shows the error rates for the SSL algorithms with respect to k on the *OSULeaf*, *Cricket-X*, and *Cricket-Y* data sets for given graph construction methods. LapRLS showed high instability with poor performance for many values of k while the other SSL algorithms showed good performance and stability on these experimental settings. Even for small variations of k , LapRLS showed high performance variation, which is not adequate for real applications. Moreover, LapRLS showed strong dependency of parameter selection with respect to k on many other experimental settings. Despite the number of theoretical studies on *manifold regularization* on *Reproducing Kernel Hilbert Spaces* (RKHS) (BELKIN; NIYOGI; SINDHWANI, 2006; SINDHWANI; NIYOGI; BELKIN, 2005; TSANG; KWOK, 2007; GENG *et al.*, 2009; SINDHWANI;

Figure 3 – Error rates for the SSL algorithms with respect to k on the *OSULeaf* (using symKNN-RBF), *Cricket-X* (using symFKNN-HM), and *Cricket-Y* (using symFKNN-RBF) data sets.

ROSENBERG, 2008), we found no theoretical evidence that could explain the ineffectiveness of LapRLS on some experimental settings.

Table 8 – Best error rates (%) for each SSL algorithm using the results shown in Fig. 3(a).

GFHF	LGC	LapRLS	RMGT
35.95	36.78	35.54	40.50

Due to the large performance variation of LapRLS, as shown in Fig. 3, we may obtain tendentious conclusions concerning the SSL algorithms’ performance in the best case analysis. For instance, suppose we are analyzing the results in Fig. 3(a) using the best case analysis, which results are reported in Table 8. If we only consider Table 8, we see that LapRLS achieved the best overall performance.

However, Fig. 3 reports that LapRLS showed high instability with poor performance for many values of k . Despite the wide use of the best case analysis in the SSL literature (ZHU, 2005; CHAPELLE; SCHÖLKOPF; ZIEN, 2006), Fig. 3(a) and Table 8 show that the use of this analysis alone may not be adequate for a comprehensive experimental analysis on graph-based SSL, possibly hiding useful information concerning the SSL algorithms’ performance with respect to graph construction and parameter selection.

Fig. 4 shows the error rates for the SSL algorithms with respect to k on the *SonyAIBORobotSurface*, *CinC-ECG-torso*, and *ECGFiveDays* data sets for given graph construction methods. RMGT showed exceptional performance and stability on these data sets, beating the other SSL algorithms by a large margin for most values of k . Probably, the effectiveness of RMGT over the other SSL algorithms on these experimental settings may be explained by the use of the normalization constraints $\mathbf{F}\mathbf{1}_c = \mathbf{1}_n$ and $\mathbf{F}^\top \mathbf{1}_n = n\omega$ into the regularization framework. Therefore, CGFHF, CLGC, and CLapRLS may also achieve exceptional performances on these experimental settings.

Identifying a priori for which data distributions or graph properties RMGT can outperform the other SSL algorithms may be relevant to practitioners. The solution of RMGT

Figure 4 – Error rates for the SSL algorithms with respect to k on the *SonyAIBORobotSurface* (using symFKNN-LLE), *CinC-ECG-torso* (using symKNN-HM), and *ECGFiveDays* (using symFKNN-HM) data sets.

can be computed with almost the same running time of the other SSL algorithms. Therefore, better results can be achieved with an insignificant increase in computation time. Moreover, RMGT is *parameter-free*⁴, which discards the necessity of parameter selection. This is a great advantage of RMGT over LapRLS, which shows strong dependency of parameter selection on some experimental settings. Unfortunately, we found no a priori knowledge that could provide insights on which data distributions or graph properties RMGT is the best suited SSL algorithm.

Figure 5 – Error rates for the SSL algorithms with respect to k on the *MedicalImages* (using symFKNN-LLE), *Wafer* (using symKNN-LLE), and *StarLightCurves* (using symKNN-RBF) data sets.

Fig. 5 shows the error rates for the SSL algorithms with respect to k on the *MedicalImages*, *Wafer*, and *StarLightCurves* data sets for given graph construction methods. RMGT showed poor performance for virtually the entire range of k values while the other SSL algorithms showed competitive performance in these data sets, which have a *high*⁵ unbalanced ratio. This uneffectiveness was observed for *all* graph construction methods. Therefore, RMGT may be

⁴ We consider k as an *external* parameter of SSL algorithms because it is associated to graph construction, not to graph regularization.

⁵ In this paper, we consider that a given data set has high unbalanced ratio if the majority class has at least *three* times more examples than the minority class.

uneffective on data sets with high unbalanced ratio. Comments on this issue can be found in Section 6.5.5.

Figure 6 – Error rates for the SSL algorithms with respect to k on the *Trace* (using symKNN-LLE), *FaceFour* (using symFKNN-RBF), and *Symbols* (using symFKNN-LLE) data sets.

Fig. 6 shows the error rates for the SSL algorithms with respect to k on the *Trace*, *FaceFour*, and *Symbols* data sets for given graph construction methods. Although the SSL algorithms achieved similar performances in the best case analysis in these experimental settings (see Tables 3 and 5), LGC showed high instability with poor performance when $k \geq 10$ while the other SSL algorithms showed similar performances for the entire range of k values. However, this *anomaly* has occurred only in a few experimental settings for certain ranges of k values. Comments on this issue can be found in Section 6.5.3.

Figure 7 – Error rates for the SSL algorithms with respect to k on the *Adiac*, *DiatomSizeReduction*, and *NonInvasiveFetalECG-Thorax2* data sets using the mutKNN-RBF graph.

Fig. 7 shows the error rates for the SSL algorithms with respect to k on the *Adiac*, *DiatomSizeReduction*, and *NonInvasiveFetalECG-Thorax2* data sets using the mutKNN-RBF graph. When RMGT shows numerical instability, LGC shows poor performance, up to 86% of error rate. We output the value -1 when an SSL algorithm found no solution for a given classification problem. Unfortunately, we found no theoretical evidence that could explain these *anomalies*, which were observed only in a few experimental settings using RBF-based graphs. Comments on this issue can be found in Section 6.5.6.

Figure 8 – Error rates for the graph construction methods on the *SonyAIBORobotSurface* (using LapRLS), *FacesUCR* (using RMGT), and *uWaveGestureLibrary-X* (using GFHF) data sets.

6.3 Evaluation of graph stability

Fig. 8 shows the error rates for the graph construction methods with respect to k on the *SonyAIBORobotSurface*, *FacesUCR*, and *uWaveGestureLibrary-X* data sets for given SSL algorithms. The mutKNN-based graphs achieved better performances than symKNN- and symFKNN-based graphs when $k \geq 16$. This superiority is highly evidenced on the *SonyAIBORobotSurface* (D31) data set for LapRLS, in which symKNN- and symFKNN-based graphs were outperformed by mutKNN-based graphs by a large margin for relatively large values of k . Moreover, mutKNN-based graphs achieved stable performances with respect to k on the *SonyAIBORobotSurface* data set.

The mutKNN-based graphs achieved better performances than symKNN- and symFKNN-based graphs with respect to k on most data sets. This effectiveness is more evidenced on unbalanced data sets. From our results and those reported by Sousa, Rezende and Batista (2013), we can suggest the use of mutKNN-based graphs instead of symKNN- and symFKNN-based graphs in real applications.

However, mutKNN-based graphs showed high instability with poor performance for $k < 10$ on the *FacesUCR* and *uWaveGestureLibrary-X* data sets. The performances for mutKNN-based graphs were satisfactory only for $k \geq 10$. Since such an instability was only evidenced in a few experimental settings, this may not be an issue on real applications. Comments on the effectiveness of mutKNN-based graphs can be found in Section 6.5.1.

Fig. 9 shows the error rates for the graph construction methods with respect to k on the *50words*, *MedicalImages*, and *WordsSynonyms* data sets for given SSL algorithms. RBF-based graphs achieved better performances than HM- and LLE-based graphs for virtually the entire range of k values. Moreover, RBF-based graphs achieved similar performances with respect to k for all adjacency graph construction methods.

Fig. 10 shows the error rates for the graph construction methods with respect to k on the *CinC-ECG-torso*, *ECGFiveDays*, and *FaceFour* data sets for given SSL algorithms. The

Figure 9 – Error rates for the graph construction methods with respect to k on the *50words* (using GFHF), *MedicalImages* (using RMGT), and *WordsSynonyms* (using GFHF) data sets.

Figure 10 – Error rates for the graph construction methods with respect to k on the *CinC-ECG-torso* (using LGC), *ECGFiveDays* (using LGC), and *FaceFour* (using LapRLS) data sets.

symKNN-HM and symFKNN-HM graphs showed poor performances for $k \geq 14$. In addition, mutKNN-HM showed no competitive performance to RBF- and LLE-based graphs for large values of k on the *ECGFiveDays* data set while achieving satisfactory performance in the other two data sets.

The ineffectiveness of symKNN-HM and symFKNN-HM for large values of k was observed only in a few experimental settings. Moreover, such a weakness was only evidenced on some data sets from ECG data and face recognition tasks. Unfortunately, we found no mathematical or statistical model in the SSL literature that could explain this issue.

Fig. 11 shows the error rates for the graph construction methods with respect to k on the *ChlorineConcentration*, *Gun-Point*, and *Coffee* data sets for given SSL algorithms. The LLE-based graphs achieved better performance and stability than HM- and RBF-based graphs for almost the entire range of k values. In addition, HM- and RBF-based graphs showed high instability and similar performances with respect to k . Comments on the effectiveness of LLE-based graphs can be found in Section 6.5.2.

Figure 11 – Error rates for the graph construction methods with respect to k on the *ChlorineConcentration* (using GFHF), *Gun-Point* (using RMGT), and *Coffee* (using LGC) data sets.

Figure 12 – Error surfaces for LGC with respect to k and μ on the *MoteStrain* (using symKNN-HM), *ItalyPowerDemand* (using symKNN-RBF), and *ChlorineConcentration* (using mutKNN-LLE) data sets.

6.4 Evaluation of error surfaces

This section analyzes the error surfaces for LGC and LapRLS with respect to their (*regularization* or *external*) parameters.

6.4.1 Error surfaces for LGC ($k \times \mu$)

Fig. 12 shows the error surfaces for LGC with respect to k and μ on the *MoteStrain*, *ItalyPowerDemand*, *ChlorineConcentration* data sets for given graph construction methods. LGC showed strong dependency of parameter selection with respect to μ . Small values of μ achieved the best results. This is in accordance with the parameter choice in the SSL literature (CHAPELLE; SCHÖLKOPF; ZIEN, 2006; ZHU, 2005; ZHOU *et al.*, 2004; JEBARA; WANG; CHANG, 2009), which states that μ should be in the set $\{0.01, 0.05\}$. Since LGC was originally formulated using *random walks* (ZHOU *et al.*, 2004), it is useful to set low values for μ .

Fig. 13 shows the error surfaces for LGC with respect to k and μ on the *FaceFour*, *Gun-Point*, and *InlineSkate* data sets for given graph construction methods. Small values of μ achieved the worst performances, which is an opposed pattern to that reported in Fig. 12.

Figure 13 – Error surfaces for LGC with respect to k and μ on the *FaceFour* (using symKNN-LLE), *Gun-Point* (using mutKNN-LLE), and *InlineSkate* (using symKNN-LLE) data sets.

Therefore, the parameter selection process for LGC with respect to k and μ may be a hard task on some experimental settings. Unfortunately, we found no evident explorable pattern that could be used for parameter selection.

Figure 14 – Error surfaces for LapRLS with respect to γ_A and γ_I on the *SonyAIBORobotSurface* (using symKNN-HM), *Gun-Point* (using symKNN-RBF), *Mallat* (using symFKNN-HM) data sets.

6.4.2 Error surfaces for LapRLS ($\gamma_A \times \gamma_I$)

Fig. 14 shows the error surfaces for LapRLS with respect to γ_A and γ_I on the *SonyAIBORobotSurface*, *Gun-Point*, *Mallat* data sets for given graph construction methods. The optimal performances only occurred when $\gamma_A = \gamma_I$. Therefore, we just need to tune one parameter value for these experimental settings without a careful parameter selection. A similar pattern was reported by [Sousa, Rezende and Batista \(2013\)](#) for a text classification task.

Fig. 15 shows the error surfaces for LapRLS on the *Symbols*, *CinC-ECG-torso*, and *Wafer* data sets for given graph construction methods. The optimal performances only occurred when $\gamma_A \neq \gamma_I$, which is an opposite pattern to that reported in Fig. 14. Therefore, the parameter selection process for LapRLS with respect to γ_A and γ_I may be a hard task on some experimental settings. We provide comments on parameter selection for LapRLS in Section 6.5.8. Unfortunately, we found no evident explorable pattern that could be used for parameter selection.

Figure 15 – Error surfaces for LapRLS with respect to γ_A and γ_I on the *Symbols* (using mutKNN-HM), *CinC-ECG-torso* (using mutKNN-RBF), and *Wafer* (using symFKNN-HM) data sets.

Figure 16 – Error surfaces for LapRLS with respect to γ_A and γ_I on the *ChlorineConcentration* (using mutKNN-RBF), *ECGFiveDays* (using mutKNN-LLE), and *SonyAIBORobotSurface* (using mutKNN-HM) data sets.

Fig. 16 shows the error surfaces for LapRLS on the *ChlorineConcentration*, *ECGFiveDays*, and *SonyAIBORobotSurface* data sets for given graph construction methods. The optimal performances occurred when $\gamma_I > \gamma_A$. This is in according with the parameter choice in the SSL literature (BELKIN; NIYOGI; SINDHWANI, 2006; MELACCI; BELKIN, 2011). We observed this behavior in at least 35% of the experimental settings.

Figure 17 – Error surfaces for LapRLS with respect to γ_A and γ_I on the *50words* (using symFKNN-HM), *Cricket-X* (using symKNN-HM), and *FaceFour* (using symKNN-LLE) data sets.

Fig. 17 shows the error surfaces for LapRLS on the *50words*, *Cricket-X*, and *FaceFour* data sets for given graph construction methods. The optimal performances only occurred when $\gamma_A > \gamma_I$, which is an opposite pattern to that reported in Fig. 16. Therefore, the parameter selection process for LapRLS with respect to γ_A and γ_I may be a hard task on some experimental settings. See Section 6.5.8 for comments on parameter selection for LapRLS.

Figure 18 – Error surfaces for LapRLS with respect to γ_A and γ_I on the *MoteStrain* (using mutKNN-RBF), *InlineSkate* (using symKNN-LLE), and *NonInvasiveFetalECG-Thorax1* (using symFKNN-LLE) data sets.

Fig. 18 shows the error surfaces for LapRLS on the *MoteStrain*, *InlineSkate*, and *NonInvasiveFetalECG-Thorax1* data sets for given graph construction methods. LapRLS achieved stable error surfaces for almost the entire parameter space. Therefore, Fig. 18 shows experimental settings in which LapRLS is weakly dependent of parameter selection with respect to γ_A and γ_I . This means that the parameter selection process for LapRLS may be unnecessary in these experimental settings.

Figure 19 – Error surfaces for LapRLS with respect to k and γ_A on the *Adiac* (using symFKNN-LLE), *ChlorineConcentration* (using mutKNN-LLE), and *FacesUCR* (using mutKNN-RBF) data sets.

6.4.3 Error surfaces for LapRLS ($k \times \gamma_A$)

Fig. 19 shows the error surfaces for LapRLS with respect to k and γ_A on the *Adiac*, *ChlorineConcentration*, and *FacesUCR* data sets for given graph construction methods. LapRLS

showed the best results for small values of γ_A . This is in accordance with the parameter choice in the SSL literature (BELKIN; NIYOGI; SINDHWANI, 2006; MELACCI; BELKIN, 2011), which states that γ_A should be small. Typically, we set $\gamma_A = 10^{-6}$ for general SSL tasks.

Figure 20 – Error surfaces for LapRLS with respect to k and γ_A on the *50words* (using symKNN-RBF), *CinC-ECG-torso* (using mutKNN-HM), and *Cricket-X* (using symKNN-LLE) data sets.

Fig. 20 shows the error surfaces for LapRLS with respect to k and γ_A on the *50words*, *CinC-ECG-torso*, and *Cricket-X* data sets for given graph construction methods. LapRLS showed poor performances for small values of γ_A and exceptional performances for the other values. This is an opposed pattern to that reported in Fig. 19. Therefore, the parameter selection process for LapRLS with respect to k and γ_A may be a hard task on some experimental settings. See Section 6.5.8 for comments on parameter selection for LapRLS.

Figure 21 – Error surfaces for LapRLS with respect to k and γ_A on the *Two-Patterns* (using symFKNN-LLE), *NonInvasiveFetalECG-Thorax1* (using symFKNN-LLE), and *MedicalImages* (using mutKNN-RBF) data sets.

Fig. 21 shows the error surfaces for LapRLS with respect to k and γ_A on the *Two-Patterns*, *NonInvasiveFetalECG-Thorax1*, and *MedicalImages* data sets for given graph construction methods. LapRLS achieved stable error surfaces for almost the entire parameter space. Therefore, Fig. 21 shows experimental settings in which LapRLS is weakly dependent of parameter selection with respect to k and γ_A . This means that the parameter selection process for LapRLS with respect to k and γ_A may be unnecessary in these experimental settings.

Figure 22 – Error surfaces for LapRLS with respect to k and γ_l on the *SonyAIBORobotSurface* (using mutKNN-RBF), *StarLightCurves* (using mutKNN-LLE), and *SwedishLeaf* (using symFKNN-RBF) data sets.

6.4.4 Error surfaces for LapRLS ($k \times \gamma_l$)

Fig. 22 shows the error surfaces for LapRLS with respect to k and γ_l on the *SonyAIBORobotSurface*, *StarLightCurves*, and *SwedishLeaf* data sets for given graph construction methods. LapRLS showed the best performances for large values of γ_l . This is in according with the parameter choice in the SSL literature (BELKIN; NIYOGI; SINDHWANI, 2006; MELACCI; BELKIN, 2011), which states that γ_l should be large. Typically, we set $\gamma_l = 10^2$ for general SSL tasks.

Figure 23 – Error surfaces for LapRLS with respect to k and γ_l on the *OSULeaf* (using symFKNN-HM), *MedicalImages* (using symFKNN-LLE), and *InlineSkate* (using symFKNN-LLE) data sets.

Fig. 23 shows the error surfaces for LapRLS with respect to k and γ_l on the *OSULeaf*, *MedicalImages*, and *InlineSkate* data sets for given graph construction methods. LapRLS showed poor performances for large values of γ_l and exceptional performances for the other values. This is an opposed pattern to that reported in Fig. 22. Therefore, the parameter selection process of LapRLS with respect to k and γ_l may be a hard task on some experimental settings. See Section 6.5.8 for comments on parameter selection for LapRLS.

Fig. 24 shows the error surfaces for LapRLS with respect to k and γ_l on the *Cricket-X*, *MoteStrain*, and *TwoLeadECG* data sets for given graph construction methods. LapRLS achieved

Figure 24 – Error surfaces for LapRLS with respect to k and γ_l on the *Cricket-X* (using symFKNN-LLE), *MoteStrain* (using symKNN-LLE), and *TwoLeadECG* (using mutKNN-RBF) data sets.

stable error surfaces for almost the entire parameter space. Therefore, Fig. 24 shows experimental settings in which LapRLS is weakly dependent of parameter selection with respect to k and γ_l . This means that the parameter selection process for LapRLS with respect to k and γ_l may be unnecessary in these experimental settings.

6.5 Discussion

This section discusses some experimental issues with their corresponding technical solutions, when available.

6.5.1 Effectiveness of mutKNN-based graphs

Table 7 shows that mutKNN-RBF achieved the best average ranking for GFHF, LGC, and LapRLS as well as the best overall average ranking. Moreover, mutKNN-based graphs achieved better performances than symKNN- and symFKNN-based graphs with respect to k on most data sets (see Fig. 8). This effectiveness is more evidenced on unbalanced data sets.

Since the set of edges of a mutKNN-based graph is a subset of the edges of a symKNN- or symFKNN-based graph for the same value of k , we suppose that mutKNN-based graphs have the advantage to remove some spurious edges between possibly dissimilar examples. Moreover, the mutKNN graph tends to connect examples within regions of constant density, but does not connect regions of different densities with each other (LUXBURG, 2007). This may be an advantage of mutKNN-based graphs over symKNN- and symFKNN-based graphs on real applications.

Many theoretical researches involving KNN-based graphs have been done in the machine learning and statistics literature (MAIER; HEIN; LUXBURG, 2009; LIU *et al.*, 2005; MAIER; LUXBURG; HEIN, 2009; MAIER; HEIN; LUXBURG, 2007; LUXBURG; ALAMGIR, 2013; ALAMGIR; LUXBURG, 2012) with many interesting properties reported. However, we found

no theoretical evidence that could explain the empirical effectiveness of mutKNN-based graphs on SSL, not restricted to unbalanced data sets.

The theory behind KNN-based graphs is still in its infancy and many statistical models should be created and/or improved in an attempt to answer this question. These models can possibly provide insights on how to create *statistically consistent* graph construction methods with respect to the training sample. However, how to formulate the optimization problem is still unclear.

6.5.2 Effectiveness of LLE-based graphs

The effectiveness of LLE-based graphs over HM- and RBF-based graphs occurs when the *reconstruction error* $\xi(\mathbf{x}_i) := \left\| \mathbf{x}_i - \sum_{\mathbf{x}_j \in \mathcal{N}_i} \mathbf{W}_{ij} \mathbf{x}_j \right\|_2^2$ is usually small for all $\mathbf{x}_i \in \mathcal{X}$. Therefore, the corresponding weighted matrix $\mathbf{W}$ is a good relational representation for the training sample $\mathcal{X}$.

There are useful methods based on LLE that can be effectively applied as well (KONG *et al.*, 2012; ZHANG; WANG, 2007; ZHAO, 2006; XIANG *et al.*, 2011; CHANG; YEUNG, 2006; LI *et al.*, 2008). For instance, Daitch, Kelner and Spielman (2009) proposed to generate $\mathbf{W}$ by solving the following optimization problem:

$$\begin{aligned} \min_{\mathbf{W} \in \mathbb{R}^{n \times n}} \sum_{i=1}^n \left\| \mathbf{D}_{ii} \mathbf{x}_i - \sum_{\mathbf{x}_j \in \mathcal{N}_i} \mathbf{W}_{ij} \mathbf{x}_j \right\|_2^2 \\ \text{s.t. } \mathbf{W} \geq 0, \mathbf{W} \mathbf{1}_n \geq \mathbf{1}_n \end{aligned} \quad (6.1)$$

This method generates a nonnegative matrix $\mathbf{W}$ with each vertex having weighted degree at least 1, i.e. $\mathbf{D}_{ii} \geq 1, \forall i \in \mathbb{N}_n$. Note that the constraint $\mathbf{W} \mathbf{1}_n \geq \mathbf{1}_n$ is equivalent to $\mathbf{D} \mathbf{1}_n \geq \mathbf{1}_n$.

The matrices generated in (6.1) are called *hard graphs*. Daitch, Kelner and Spielman (2009) also proposed the concept of *θ -soft graphs*, which are generated in (6.1), substituting the constraint $\mathbf{D} \mathbf{1}_n \geq \mathbf{1}_n$ by the following constraint:

$$\sum_{i=1}^n (\max(0, 1 - \mathbf{D}_{ii}))^2 \leq \theta n$$

This new constraint is used to deal with *outliers* in the training sample, allowing some vertices (supposed to be outliers) to have lower weighted degree $\mathbf{D}_{ii}$.

Despite the effectiveness of LLE on some experimental settings, identifying for which type of data LLE-based graphs can effectively outperform HM- and RBF-based graphs is still an open problem. This may be of great interest to practitioners since we do not have parameters to tune and there exist efficient solvers to minimize the reconstruction error for each $\mathbf{x}_i$ subject to $\mathbf{W} \mathbf{1}_n = \mathbf{1}_n$ and $\mathbf{W} \geq 0$ (LIU; HE; CHANG, 2010; KONG *et al.*, 2012).

6.5.3 Uneffectiveness of LGC

Fig. 7 shows the uneffectiveness of LGC on some experimental settings for many values of k . We may solve this problem for LGC by applying the fitting term in the objective function only on labeled examples. Specifically, we set $\Sigma_{\mathcal{L}} = \mu \mathbf{I}_l$ and $\Sigma_{\mathcal{U}} = \mathbf{O}_{u \times u}$. We can also incorporate in the objective function of LGC the node regularizer $\mathbf{V} \in \mathbb{R}^{n \times n}$ proposed by Wang *et al.* (2008). This yields the following optimization problem:

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

such that $\mathbf{V} := \text{diag}(\mathbf{v})$ is defined by:

$$\mathbf{v} = \sum_{j=1}^c \frac{\mathbf{Y}_{:,j} \odot \mathbf{D} \mathbf{1}_n}{\mathbf{Y}_{:,j}^\top \mathbf{D} \mathbf{1}_n}$$

We obtain the following closed-form solution:

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

This simple approach solves the uneffectiveness of LGC on some experimental settings. However, this would lead to poor results when the degree distribution for labeled examples is so far from the uniform distribution.

6.5.4 Adaptive kernel size for RMGT

Fig. 5 and Tables 3, 4, and 5 show that RMGT achieved poor performance on some experimental settings. Probably, a global kernel size may not be adequate for RMGT in these experimental settings.

We can solve this problem using adaptive kernel sizes σ_{ij}^k on the RBF kernel, e.g. using the HM method. Other strategies may also be applied as well (ZELNIK-MANOR; PERONA, 2005; YANG *et al.*, 2008; WANG *et al.*, 2008). For instance, Zelnik-Manor and Perona (2005) estimate the σ_{ij}^k 's as follows:

$$\sigma_{ij}^k = \sqrt{\frac{\Psi(\mathbf{x}_i, \mathbf{x}_i^{(k)}) \Psi(\mathbf{x}_j, \mathbf{x}_j^{(k)})}{2}}$$

Yang *et al.* (2008) estimate the σ_{ij}^k 's as follows:

$$\sigma_{ij}^k = r \left(\frac{\Psi(\mathbf{x}_i, \mathbf{x}_i^{(k)}) + \Psi(\mathbf{x}_j, \mathbf{x}_j^{(k)})}{2} \right) \quad (6.2)$$

where $r \in \mathbb{R}_+^*$ is a free-parameter. Wang *et al.* (2008) achieved satisfactory results in their experiments using the adaptive kernel sizes in (6.2) with $r = 1$.

Despite the increasing popularity of adaptive weight methods based on the RBF kernel (ZELNIK-MANOR; PERONA, 2005; YANG *et al.*, 2008; WANG *et al.*, 2008), it is still unclear for which type of data a given adaptive kernel size is the best suited. We believe that many heuristic methods can be created to estimate the σ_{ij}^k 's based on the distances $\Psi(\mathbf{x}_i, \mathbf{x}_i^{(k)})$ and $\Psi(\mathbf{x}_j, \mathbf{x}_j^{(k)})$. We also encourage researchers with strong mathematical background to provide adaptive weight methods based on existing ones (ZHU; GHAHRAMANI; LAFFERTY, 2003; KAPOOR *et al.*, 2005; ZHANG; LEE, 2007).

6.5.5 Uneffectiveness of RMGT on data sets with high unbalanced ratio

Fig. 5 shows that RMGT achieved poor performances for *all* values of k . We observed this result on many unbalanced data sets, specially those with high unbalanced ratio. Unfortunately, we found no approach to solve this problem.

Probably, by using class prior knowledge combined with some class mass normalization procedure (e.g. (ZHU; GHAHRAMANI; LAFFERTY, 2003)), RMGT might achieve better performances on data sets with high unbalanced ratio. Moreover, existing algorithms for dealing with unbalanced data sets (HE; GARCIA, 2000) might be combined with RMGT in an attempt to overcome this problem.

6.5.6 Numerical instabilities of RMGT

Tables 3, 4, and 5 show that RMGT found no solution for the *DiatomSizeReduction* (D11), *Fish* (D17), and *NonInvasiveFetalECG-Thorax2* (D28) data sets using RBF-based graphs for *all* values of k . In addition, for the *Adiac* (D2) and *Wafer* (D43) data sets, RMGT found no solution using RBF-based graphs for many values of k .

In these cases, $(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}$ is a *singular* matrix, which means that a linear system $(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}\mathbf{a} = \mathbf{b}$ has no solution, independently of the vector $\mathbf{b} \in \mathbb{R}^u$. Therefore, RMGT found no solution for the classification problem. Since we used the normalized Laplacian $\mathbf{L}_{\mathbb{N}}$ instead of the combinatorial Laplacian $\mathbf{L}_{\mathbb{C}}$ in GFHF, this classifier showed no numerical instability in our experiments. This means that CGFHF, CLGC, and CLapRLS would find a good solution for these experimental settings using the normalized Laplacian.

6.5.7 Effectiveness of LapRLS

Table 6 shows that LapRLS achieved the best average ranking for *all* graph construction methods as well as the best overall average ranking. Moreover, Tables 3, 4, and 5 show that

LapRLS achieved competitive performances on most experimental settings. Despite its impressive empirical effectiveness in the best case analysis, LapRLS has two regularization parameters and the parameter selection process for this method is still unclear.

Unfortunately, we found no mathematical or statistical model in the SSL literature that could explain the effectiveness of LapRLS on the experimental settings reported in Fig. 2. Although there is high theoretical motivation to use such a method (SINDHWANI; NIYOGI; BELKIN, 2005; BELKIN; NIYOGI; SINDHWANI, 2006), a good mathematical model does not imply good empirical performance for a given application.

Finding an answer that explains the effectiveness of LapRLS may help practitioners identifying *a priori* for which data distributions or graph properties LapRLS may have the best performance and stability over the other SSL algorithms with respect to k . Although there are dozens of graph properties in the complex network literature (COSTA *et al.*, 2007) (e.g. *entropy of the degree distribution*, *betweenness centrality*, and *closeness*), it is unclear which of them can be effectively used as meta-features in an attempt to solve this problem.

Unfortunately, we found no mathematical model (or even an intuitive idea) in the SSL literature that could solve this problem, even in a purely heuristic way. Therefore, the practitioners have no tools to make an informed choice of values for the regularization parameters of LapRLS.

6.5.8 Parameter selection for LapRLS

The regularization framework of LapRLS can be viewed as a natural extension of many existing graph-based SSL algorithms (BELKIN; NIYOGI; SINDHWANI, 2006), providing kernel expansions over both labeled and unlabeled examples. Therefore, finding a way to provide parameter selection for this method can provide insights on parameter selection for many other graph-based SSL algorithms.

We observed a significant performance variation in the error surfaces for LapRLS on many experimental settings. Unfortunately, these observations may hinder the parameter selection process for this method.

6.5.9 Kernel selection for LapRLS

In our experiments, we generated the kernel matrix $\mathbf{K}$ using the RBF kernel with a heuristic chosen value for σ , as suggested by Jebara, Wang and Chang (2009). Since the RBF kernel may not be the best suited kernel function for all data sets⁶, we have to choose a kernel function with corresponding parameter values that can effectively represent the intrinsic

⁶ We found no experimental analyses in the machine learning literature that could lead to useful insights on which kernel function is the best suited for a given data distribution. Such an experimental study can be of great interest to practitioners.

geometric properties of the data. For instance, we can use the *polynomial* and *sigmoidal* kernel functions, $K_{\text{pol}}(\cdot, \cdot)$ and $K_{\text{sig}}(\cdot, \cdot)$, which are defined as follows:

$$K_{\text{pol}}(\mathbf{x}_i, \mathbf{x}_j) = (a\langle \mathbf{x}_i, \mathbf{x}_j \rangle_n + b)^q, \quad K_{\text{sig}}(\mathbf{x}_i, \mathbf{x}_j) = \tanh(a\langle \mathbf{x}_i, \mathbf{x}_j \rangle + b)$$

where $\langle \cdot, \cdot \rangle_n$ represents the inner product of vectors in $\mathbb{R}^n$. $a, b \in \mathbb{R}$ and $q \in \mathbb{N}^*$ are the parameters of the kernel functions.

There are dozens of other kernel functions or kernel selection methods that can be effectively applied on real applications of SSL (DAI; YEUNG, 2007; ZHU *et al.*, 2005; ARGYRIOU; HERBSTER; PONTIL, 2006; ARGYRIOU; MICCHELLI; PONTIL, 2005; MICCHELLI; PONTIL, 2005). For instance, Argyriou, Micchelli and Pontil (2005) and Micchelli and Pontil (2005) learn the kernel function in a nonparametric manner by joint minimization over regularization frameworks. Zhu *et al.* (2005) and Argyriou, Herbster and Pontil (2006) learn an optimal kernel function from regularization using a graph Laplacian $\mathbf{L}$ to capture the geometric structure of the data. Unfortunately, to our knowledge, there is no way to identify *a priori* which kernel function or kernel selection method may be the best suited for a given data distribution.

6.6 Our findings

This chapter showed our experimental analysis on time series transductive classification on graphs. Our findings are summarized as follows:

1. RMGT showed numerical instabilities on some data sets using RBF-based graphs. These instabilities have occurred because $(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}$ was singular. Probably, a global kernel size may not be adequate for these experimental settings when using RMGT. We suggested the use of adaptive kernel sizes to overcome this problem;
2. when RMGT showed numerical instability, LGC showed poor performance. Moreover, LGC showed poor performance for large values of k on some experimental settings. We can use the approaches in Section 6.5.3 to overcome this problem;
3. LLE-based graphs outperformed HM- and RBF-based graphs on some experimental settings. This effectiveness has occurred because the local reconstruction error on each $\mathbf{x}_i \in \mathcal{X}$ is small, which means that the weighted matrix $\mathbf{W}$ is a good relational representation of the training sample for these experimental settings. Many LLE-based methods can also be effectively applied on these experimental settings as well (DAITCH; KELNER; SPIELMAN, 2009; KONG *et al.*, 2012; ZHANG; WANG, 2007; ZHAO, 2006; XIANG *et al.*, 2011; CHANG; YEUNG, 2006; LI *et al.*, 2008);
4. LapRLS achieved the best average ranking for *all* graph construction methods as well as the best overall average ranking in the best case analysis. This method statistically outper-

formed the competing SSL algorithms for most graph construction methods. Therefore, for the best case analysis, LapRLS is the most effective SSL algorithm;

5. mutKNN-RBF achieved the best average ranking for most SSL algorithms as well as the best overall average ranking. Therefore, this is the best suited graph construction method for the best case analysis on general SSL tasks. This is in according with the results shown in (SOUSA; REZENDE; BATISTA, 2013);
6. mutKNN-based graphs achieved better performances than symKNN- and symFKNN-based graphs on most experimental settings. However, mutKNN-based graphs showed high instability for small values of k on a few experimental settings. This is in according with the results shown in (SOUSA; REZENDE; BATISTA, 2013);
7. as shown in Fig. 3(a) and Table 8, the use of the best case analysis alone may not be adequate for a comprehensive experimental evaluation on graph-based SSL. This analysis may hide useful information concerning the classifiers' performance with respect to graph construction and parameter selection;
8. RMGT showed exceptional performance and stability with respect to k on some experimental settings. Probably, such an effectiveness over the other SSL algorithms on these settings may be explained by the use of the normalization constraints $\mathbf{F}\mathbf{1}_c = \mathbf{1}_n$ and $\mathbf{F}^\top \mathbf{1}_n = n\omega$ into the regularization framework, as discussed by Sousa, Rezende and Batista (2013);
9. RBF-based graphs achieved the best average rankings for the GFHF, LGC, and LapRLS algorithms. Therefore, the RBF kernel is the best suited weighted matrix generation method for most SSL algorithms in the best case analysis. This is in according with the results shown in (SOUSA; REZENDE; BATISTA, 2013).

EXPERIMENTS ON CONSTRAINED GRAPH-BASED SEMI-SUPERVISED LEARNING

In this chapter, we provide our experimental analysis on constrained graph-based SSL. Specifically, we compare the performances of the constrained methods proposed in Chapter 4 with the performances reported by Sousa, Rezende and Batista (2013) for GFHF, LGC, LapRLS, LapSVM, and RMGT. We used the experimental setup described in Chapter 5 and the empirical evaluation models described in Section 5.5.

This chapter contains the results in (SOUSA; BATISTA, 2015; SOUSA; BATISTA, 2016) with additional experimental results. The document containing extensive experimental results on constrained graph-based SSL is freely available¹. The data sets and source codes used in these experiments are also freely available².

We organized this chapter as follows. Section 7.1 analyzes the results for the best case analysis. Section 7.2 analyzes the results for the evaluation of classifier stability. Section 7.3 analyzes the results for the evaluation of error surfaces. Finally, Section 7.4 describes our findings for this set of experiments.

7.1 Best case analysis

Tables 9 to 17 show the best average error rates with standard deviations for the SSL algorithms using given graph construction methods in the partitions of 10 and 100 labeled examples. In each table, we also report the average ranking of each SSL algorithm over all data

¹ <https://www.researchgate.net/publication/316634288_Constrained_graph-based_semi-supervised_learning_supplementary_material>.

² <http://sites.labc.icmc.usp.br/sousa/constrained_graph_SSL/>.

sets. The three worst performances on each data set have a grey background while the best one is in bold. The rankings for the algorithms that were statistically outperformed by the best ranked algorithm have a grey background. As in Chapter 6, we used Friedman’s test (FRIEDMAN, 1940) with Nemenyi’s post test (NEMENYI, 1963) with a significance level of 0.05.

Table 9 – Best average error rates (%) and standard deviations (%) for the SSL algorithms using symKNN-RBF.

$l = 10$	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)	6.5
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)	5.5
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)	4.833
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)	6.167
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)	5.167
CGFHF	13.32 (3.64)	30.98 (3.52)	7.42 (3.37)	36.11 (16.00)	26.94 (4.95)	30.22 (2.47)	3.5
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)	2.667
CLapRLS	11.39 (3.72)	30.37 (4.44)	5.30 (2.29)	33.32 (6.54)	26.87 (5.00)	28.45 (3.73)	1.667
$l = 100$	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)	6.167
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)	6.0
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)	4.417
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)	4.417
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)	5.833
CGFHF	3.80 (1.05)	3.09 (1.79)	2.14 (0.40)	9.95 (0.87)	18.56 (0.93)	21.17 (0.92)	4.333
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)	2.583
CLapRLS	3.07 (0.43)	2.76 (1.40)	1.99 (0.51)	10.00 (1.40)	17.17 (0.93)	20.67 (1.11)	2.25

Table 10 – Best average error rates (%) and standard deviations (%) for the SSL algorithms using mutKNN-RBF.

$l = 10$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	9.75 (4.50)	35.07 (3.82)	9.35 (4.51)	46.93 (4.81)	46.72 (4.85)	37.51 (6.85)	6.75
LGC	9.93 (4.34)	35.07 (3.82)	10.54 (5.21)	39.81 (5.36)	41.85 (4.32)	34.78 (6.55)	6.25
LapRLS	9.75 (4.53)	33.56 (7.32)	9.33 (4.48)	38.36 (5.96)	40.66 (5.45)	34.58 (6.14)	4.167
LapSVM	9.91 (2.51)	34.37 (6.47)	8.67 (3.89)	38.90 (6.49)	40.90 (6.08)	37.49 (7.06)	5.167
RMGT	13.08 (3.41)	28.95 (3.88)	8.13 (3.14)	46.11 (4.50)	42.76 (6.11)	27.77 (5.95)	4.833
CGFHF	10.88 (4.11)	27.80 (5.10)	6.38 (2.37)	36.83 (12.12)	33.26 (4.41)	28.74 (4.01)	3.0
CLGC	10.88 (4.14)	29.18 (3.40)	5.46 (2.86)	36.88 (12.11)	32.92 (4.75)	28.56 (3.48)	3.333
CLapRLS	10.90 (4.62)	29.17 (3.39)	5.29 (3.17)	34.12 (10.43)	32.30 (4.49)	28.44 (3.89)	2.5
$l = 100$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	3.33 (1.05)	1.09 (1.59)	2.17 (0.37)	32.56 (5.28)	38.62 (5.58)	23.69 (2.48)	6.417
LGC	3.58 (0.99)	1.09 (1.59)	2.43 (0.43)	26.20 (2.58)	34.45 (2.89)	23.35 (2.26)	6.083
LapRLS	3.28 (0.98)	0.39 (0.92)	2.05 (0.41)	23.92 (2.14)	28.73 (3.43)	23.48 (1.64)	4.0
LapSVM	3.71 (1.44)	0.55 (1.01)	2.13 (0.41)	21.64 (3.90)	25.51 (5.58)	22.43 (1.10)	4.167
RMGT	4.84 (1.83)	1.95 (1.59)	2.31 (0.37)	31.08 (5.09)	33.12 (3.51)	20.89 (1.07)	6.0
CGFHF	3.31 (0.85)	2.58 (1.43)	2.17 (0.30)	15.82 (1.17)	26.48 (1.99)	21.09 (0.90)	4.667
CLGC	2.87 (0.80)	2.44 (1.50)	1.97 (0.40)	15.01 (1.27)	24.59 (1.80)	20.60 (1.01)	2.333
CLapRLS	2.86 (0.80)	2.26 (1.33)	1.98 (0.38)	15.32 (1.33)	21.70 (1.93)	20.61 (1.01)	2.333

CGFHF, CLGC, and CLapRLS showed good to exceptional performances on most data sets. In addition, they achieved the best average rankings in both partitions with CLapRLS being the best ranked method in both partitions for most graph construction methods.

On USPS, the proposed methods showed competitive performances to the other SSL algorithms for *all* graph construction methods in both partitions. In addition, they showed the

Table 11 – Best average error rates (%) and standard deviations (%) for the SSL algorithms using symFKNN-RBF.

$l = 10$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	10.75 (3.77)	35.22 (6.92)	10.01 (3.93)	46.12 (7.65)	46.34 (6.83)	38.49 (5.87)	6.333
LGC	10.97 (3.00)	34.81 (6.22)	10.47 (4.66)	37.95 (6.66)	40.09 (5.46)	35.51 (5.64)	5.333
LapRLS	10.57 (2.90)	35.50 (5.84)	10.02 (3.92)	38.08 (6.64)	40.36 (6.02)	35.34 (5.73)	5.333
LapSVM	11.04 (3.43)	34.04 (6.91)	9.47 (4.19)	39.16 (6.07)	40.91 (6.08)	39.45 (6.30)	5.833
RMGT	16.02 (2.85)	32.94 (4.20)	8.55 (3.36)	45.25 (6.07)	38.31 (6.02)	29.65 (6.46)	5.167
CGFHF	12.54 (3.32)	30.89 (3.53)	7.00 (2.74)	36.86 (15.34)	26.79 (4.94)	29.42 (2.31)	3.5
CLGC	12.21 (3.42)	30.62 (5.19)	5.68 (2.63)	36.88 (15.32)	26.64 (4.86)	28.03 (2.46)	2.75
CLapRLS	11.34 (3.40)	30.31 (4.38)	5.17 (2.37)	33.51 (8.18)	26.64 (4.86)	27.61 (2.23)	1.75
$l = 100$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	3.26 (1.00)	1.14 (1.46)	2.19 (0.48)	32.17 (5.70)	37.59 (5.75)	24.12 (3.13)	6.25
LGC	3.54 (0.96)	1.14 (1.48)	2.45 (0.61)	22.81 (2.48)	30.42 (2.46)	23.59 (1.42)	5.917
LapRLS	3.17 (0.93)	0.78 (1.13)	2.14 (0.37)	22.74 (1.79)	28.73 (3.43)	23.65 (1.66)	4.25
LapSVM	3.94 (1.73)	0.79 (1.13)	2.14 (0.39)	21.64 (3.90)	25.51 (5.58)	23.09 (1.81)	4.25
RMGT	6.15 (2.10)	1.72 (1.63)	2.35 (0.39)	28.40 (8.28)	27.02 (2.25)	20.76 (1.18)	5.833
CGFHF	3.67 (1.09)	2.93 (1.90)	2.15 (0.37)	10.56 (0.87)	19.40 (1.09)	21.09 (0.97)	4.833
CLGC	3.06 (0.67)	2.70 (1.24)	1.97 (0.50)	10.26 (1.18)	17.66 (0.89)	20.23 (1.03)	2.583
CLapRLS	2.99 (0.60)	2.62 (1.23)	1.97 (0.50)	10.26 (1.18)	17.66 (0.89)	20.23 (1.02)	2.083

Table 12 – Best average error rates (%) and standard deviations (%) for the SSL algorithms using symKNN-HM.

$l = 10$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	15.53 (2.76)	38.55 (6.06)	10.73 (4.27)	46.86 (5.28)	46.19 (7.25)	42.31 (8.54)	7.333
LGC	14.49 (5.25)	37.20 (7.32)	11.53 (5.00)	38.36 (6.83)	40.27 (4.48)	37.51 (4.48)	5.667
LapRLS	14.56 (3.89)	37.58 (5.91)	10.75 (4.24)	38.18 (6.70)	40.24 (6.07)	37.12 (4.52)	5.5
LapSVM	14.63 (5.47)	36.40 (4.07)	10.13 (3.65)	39.15 (6.04)	40.91 (6.08)	43.06 (4.99)	6.167
RMGT	19.08 (1.21)	31.20 (6.14)	8.07 (2.69)	44.31 (9.03)	38.48 (6.91)	34.86 (6.04)	5.167
CGFHF	14.51 (4.87)	28.74 (6.52)	6.03 (1.96)	35.95 (15.70)	26.59 (4.53)	30.31 (3.23)	3.0
CLGC	14.33 (4.54)	27.29 (6.86)	5.50 (2.57)	35.97 (15.70)	26.38 (4.30)	27.77 (3.81)	2.083
CLapRLS	13.25 (3.63)	27.29 (6.86)	4.61 (1.47)	32.70 (6.62)	26.36 (4.32)	27.48 (3.14)	1.083
$l = 100$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	6.30 (2.30)	3.80 (1.74)	2.01 (0.58)	31.75 (5.79)	37.78 (5.59)	28.48 (3.08)	7.5
LGC	5.66 (0.96)	4.10 (1.96)	2.31 (0.52)	22.09 (1.65)	29.81 (2.57)	27.71 (2.84)	6.833
LapRLS	5.32 (1.10)	3.34 (1.52)	1.87 (0.51)	21.89 (1.73)	28.72 (3.43)	27.20 (2.97)	4.667
LapSVM	5.74 (1.38)	3.34 (1.58)	1.92 (0.48)	21.64 (3.90)	25.51 (5.58)	27.67 (2.57)	5.167
RMGT	8.31 (2.37)	2.93 (1.19)	1.95 (0.40)	26.27 (8.72)	25.11 (2.12)	24.73 (1.36)	5.167
CGFHF	4.37 (0.95)	2.91 (1.17)	1.93 (0.41)	9.62 (0.94)	18.23 (0.83)	22.54 (1.21)	2.833
CLGC	3.98 (1.03)	3.05 (1.58)	1.64 (0.48)	9.57 (1.10)	17.05 (0.83)	22.30 (1.34)	1.667
CLapRLS	3.98 (1.03)	2.97 (1.62)	1.58 (0.46)	9.73 (1.17)	17.06 (0.80)	22.30 (1.25)	2.167

best performances on USPS for the partitions of 100 labeled examples. Since CGFHF, CLGC, and CLapRLS generalize RMGT, we would expect that they inherit its weaknesses. Specifically, since RMGT may be ineffective on data sets with *high unbalanced ratio* (see Section 6.5.5), we expect that the proposed constrained methods would also be ineffective on this scenario.

However, such a weakness was evidenced only in a few experimental settings, usually for large values of k . Probably, the use of normalized Laplacians combined with higher order regularization in the proposed constrained methods overcame this weakness for most experimental settings. Comprehensive experiments on more data sets with high unbalanced ratio are needed to verify the validity of this hypothesis.

Table 13 – Best average error rates (%) and standard deviations (%) for the SSL algorithms using mutKNN-HM.

$l = 10$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	11.01 (3.59)	35.30 (3.87)	10.02 (6.36)	46.66 (6.18)	46.58 (5.06)	41.18 (9.87)	7.5
LGC	10.79 (3.75)	35.19 (4.89)	10.96 (5.34)	39.51 (5.80)	41.94 (4.19)	36.01 (5.63)	6.667
LapRLS	10.57 (4.66)	32.80 (7.67)	9.92 (5.50)	38.29 (6.00)	40.66 (5.50)	35.90 (5.61)	4.833
LapSVM	10.04 (2.83)	33.08 (6.35)	9.58 (4.73)	39.00 (6.43)	40.90 (6.08)	42.10 (6.03)	5.5
RMGT	16.98 (2.45)	28.00 (4.67)	7.50 (2.43)	44.73 (5.48)	40.53 (4.37)	31.12 (6.35)	4.833
CGFHF	10.60 (4.18)	27.69 (5.58)	5.22 (1.77)	35.20 (12.80)	34.12 (4.51)	29.00 (3.57)	2.833
CLGC	9.92 (3.24)	29.17 (4.71)	5.04 (2.93)	35.21 (12.80)	34.03 (4.39)	28.49 (3.52)	2.5
CLapRLS	8.71 (3.20)	28.94 (5.43)	4.10 (1.77)	33.49 (9.93)	33.89 (4.34)	27.42 (3.53)	1.333
$l = 100$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	3.78 (1.32)	2.27 (1.70)	2.09 (0.53)	32.42 (5.29)	38.96 (6.06)	25.29 (2.40)	7.167
LGC	3.97 (0.87)	2.22 (1.52)	2.35 (0.62)	26.17 (3.01)	34.70 (2.74)	25.01 (2.59)	6.667
LapRLS	3.73 (1.25)	0.61 (0.88)	2.03 (0.38)	23.95 (2.13)	28.73 (3.43)	24.84 (1.90)	4.333
LapSVM	3.71 (1.26)	0.65 (0.90)	2.02 (0.40)	21.64 (3.90)	25.51 (5.58)	24.66 (1.62)	3.333
RMGT	4.71 (2.25)	2.41 (1.49)	2.06 (0.49)	29.64 (6.28)	32.02 (2.78)	22.48 (1.08)	6.333
CGFHF	3.18 (0.66)	2.46 (1.44)	2.04 (0.34)	15.80 (1.48)	27.29 (1.98)	22.18 (0.83)	4.333
CLGC	2.97 (0.77)	2.09 (1.19)	1.85 (0.49)	14.93 (1.33)	25.65 (1.80)	21.92 (0.64)	2.167
CLapRLS	2.98 (0.77)	1.77 (1.19)	1.83 (0.54)	15.10 (1.31)	21.70 (1.93)	21.91 (0.64)	1.667

Table 14 – Best average error rates (%) and standard deviations (%) for the SSL algorithms using symFKNN-HM.

$l = 10$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	15.17 (3.09)	37.77 (6.25)	10.24 (4.36)	46.77 (5.38)	46.27 (7.05)	42.20 (8.62)	7.333
LGC	14.62 (3.33)	36.36 (8.25)	11.07 (4.76)	38.13 (7.11)	40.17 (4.75)	37.49 (4.35)	6.0
LapRLS	14.38 (4.14)	36.93 (4.95)	10.28 (4.32)	38.06 (6.52)	40.11 (6.06)	37.32 (4.38)	5.5
LapSVM	14.35 (4.28)	36.57 (3.57)	9.93 (3.95)	39.14 (6.05)	40.91 (6.08)	42.71 (5.15)	6.0
RMGT	18.88 (2.26)	30.56 (5.52)	7.92 (2.58)	44.68 (7.89)	38.48 (6.67)	34.61 (6.25)	5.167
CGFHF	13.78 (4.69)	28.34 (7.88)	5.89 (2.00)	36.62 (15.29)	26.76 (4.96)	28.65 (2.68)	2.833
CLGC	13.77 (4.69)	27.28 (6.67)	5.33 (2.04)	36.64 (14.97)	26.44 (4.92)	26.79 (3.07)	2.0
CLapRLS	12.87 (4.25)	27.28 (6.67)	4.35 (1.30)	33.04 (8.37)	26.44 (4.92)	26.65 (2.82)	1.167
$l = 100$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	5.92 (2.19)	3.02 (1.64)	2.04 (0.52)	31.71 (6.08)	37.68 (5.60)	27.62 (3.87)	7.5
LGC	5.40 (0.74)	3.52 (2.08)	2.12 (0.72)	22.04 (1.75)	30.21 (3.27)	26.78 (1.37)	6.833
LapRLS	4.87 (1.04)	2.62 (1.43)	1.95 (0.55)	21.99 (1.75)	28.72 (3.43)	26.82 (1.83)	5.167
LapSVM	5.30 (1.19)	2.72 (1.56)	1.93 (0.54)	21.64 (3.90)	25.51 (5.58)	26.50 (2.13)	4.667
RMGT	7.71 (2.52)	2.11 (1.05)	1.97 (0.52)	26.01 (8.82)	25.49 (2.21)	23.47 (1.06)	5.25
CGFHF	4.04 (0.88)	2.26 (1.03)	1.97 (0.53)	9.64 (0.78)	18.79 (0.93)	22.05 (0.98)	3.583
CLGC	3.49 (0.72)	2.08 (1.31)	1.70 (0.49)	9.49 (1.11)	17.31 (0.95)	21.44 (1.11)	1.583
CLapRLS	3.49 (0.72)	2.02 (1.45)	1.65 (0.45)	9.56 (1.04)	17.30 (0.92)	21.61 (0.94)	1.417

On COIL₂, the proposed methods showed better performances than the other SSL algorithms for *all* graph construction methods for the partitions of 10 labeled examples. Moreover, they showed the best performances in the partitions of 100 labeled examples for LLE-based graphs. However, our methods showed no competitive performances in the partitions of 100 labeled examples for RBF-based graphs and mutKNN-HM.

On DIGIT-1, the proposed methods showed better performances than the other SSL algorithms for *all* graph construction methods in both partitions. Moreover, they outperformed the unconstrained methods by a large margin in the partitions of 10 labeled examples.

On G-241C for both partitions, the proposed methods outperformed the competing

Table 15 – Best average error rates (%) and standard deviations (%) for the SSL algorithms using symKNN-LLE.

$l = 10$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	16.03 (2.47)	36.04 (5.60)	10.94 (4.69)	47.54 (3.77)	47.33 (4.77)	43.56 (6.96)	7.5
LGC	15.05 (4.33)	35.95 (6.08)	11.49 (5.41)	41.22 (4.12)	43.33 (3.03)	39.18 (4.02)	6.167
LapRLS	14.73 (3.24)	36.85 (5.25)	10.93 (4.66)	38.68 (5.60)	40.61 (5.51)	38.49 (4.00)	4.667
LapSVM	14.81 (3.38)	35.39 (4.79)	10.31 (4.11)	39.12 (6.43)	40.90 (6.07)	42.77 (6.20)	5.0
RMGT	19.04 (1.19)	30.63 (3.94)	7.91 (2.49)	42.83 (6.00)	42.25 (3.32)	36.60 (4.79)	5.5
CGFHF	14.77 (4.11)	29.79 (5.70)	6.30 (2.02)	41.54 (9.67)	36.14 (3.92)	32.13 (3.56)	3.667
CLGC	14.65 (4.06)	28.96 (5.22)	5.35 (2.44)	41.50 (9.71)	35.45 (4.05)	30.16 (3.64)	2.333
CLapRLS	13.45 (3.72)	29.05 (5.06)	4.45 (1.48)	36.39 (5.67)	35.44 (4.06)	29.08 (2.41)	1.167
$l = 100$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	7.44 (2.35)	2.06 (1.79)	2.18 (0.53)	33.13 (4.39)	38.65 (4.76)	28.79 (3.06)	7.5
LGC	6.45 (0.77)	2.36 (1.97)	2.37 (0.91)	29.06 (1.74)	34.36 (2.56)	28.24 (2.97)	7.0
LapRLS	5.98 (1.22)	1.69 (1.30)	2.12 (0.57)	23.96 (2.17)	28.73 (3.44)	27.20 (2.93)	5.167
LapSVM	6.37 (0.97)	1.68 (1.43)	2.08 (0.46)	21.64 (3.90)	25.52 (5.58)	28.55 (2.45)	4.833
RMGT	9.59 (2.52)	1.42 (1.02)	2.07 (0.39)	28.64 (3.14)	31.64 (1.69)	24.29 (1.53)	5.167
CGFHF	4.21 (0.65)	0.80 (0.69)	2.08 (0.41)	20.90 (1.59)	26.45 (1.28)	22.87 (1.17)	3.333
CLGC	3.85 (0.56)	0.59 (0.76)	1.60 (0.30)	19.43 (1.36)	24.28 (0.85)	21.35 (1.39)	1.667
CLapRLS	3.90 (0.68)	0.52 (0.64)	1.54 (0.26)	18.88 (2.60)	21.70 (1.93)	21.39 (1.09)	1.333

Table 16 – Best average error rates (%) and standard deviations (%) for the SSL algorithms using mutKNN-LLE.

$l = 10$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	11.64 (3.38)	35.20 (3.79)	10.30 (5.89)	47.14 (2.87)	46.98 (3.64)	42.34 (6.48)	7.667
LGC	11.04 (3.82)	35.18 (3.76)	10.96 (6.34)	42.12 (3.90)	42.94 (3.09)	37.54 (5.41)	6.5
LapRLS	11.28 (4.09)	31.78 (7.81)	10.19 (5.92)	38.66 (5.72)	40.59 (5.61)	37.28 (5.26)	4.667
LapSVM	10.61 (2.49)	31.54 (6.24)	10.22 (5.52)	38.95 (6.46)	40.82 (6.66)	41.80 (7.65)	4.667
RMGT	17.85 (1.95)	29.48 (4.16)	7.53 (2.11)	43.75 (6.40)	42.12 (4.08)	33.89 (5.32)	5.5
CGFHF	10.69 (3.36)	27.30 (5.56)	5.46 (1.84)	41.38 (6.58)	39.84 (5.05)	28.71 (3.13)	3.0
CLGC	10.68 (3.37)	28.75 (4.20)	4.96 (1.67)	41.38 (6.58)	39.03 (4.94)	27.18 (2.59)	2.833
CLapRLS	9.59 (4.16)	28.66 (4.21)	4.21 (1.54)	36.61 (4.67)	37.67 (4.37)	27.10 (2.62)	1.167
$l = 100$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	4.48 (1.55)	1.34 (1.49)	2.21 (0.64)	33.78 (4.64)	39.12 (3.65)	25.86 (2.37)	7.5
LGC	4.34 (0.81)	1.43 (1.55)	2.31 (0.72)	30.22 (2.11)	36.13 (2.40)	25.49 (1.80)	7.0
LapRLS	4.14 (0.88)	0.35 (0.75)	2.14 (0.44)	23.96 (2.18)	28.73 (3.43)	25.33 (1.66)	4.167
LapSVM	4.18 (1.19)	0.36 (0.74)	2.11 (0.41)	21.64 (3.90)	25.51 (5.58)	25.57 (2.40)	3.833
RMGT	5.84 (2.39)	0.97 (0.76)	2.11 (0.29)	29.93 (4.80)	34.09 (2.10)	22.73 (1.09)	5.833
CGFHF	3.86 (0.87)	0.37 (0.49)	2.06 (0.30)	23.67 (1.30)	32.40 (2.18)	21.93 (0.78)	3.833
CLGC	3.55 (0.92)	0.37 (0.43)	1.65 (0.39)	21.85 (1.07)	31.40 (2.82)	21.10 (0.66)	2.5
CLapRLS	3.47 (0.75)	0.23 (0.43)	1.68 (0.30)	18.91 (2.69)	21.70 (1.93)	21.18 (0.62)	1.333

methods by a large margin for symKNN- and symFKNN-based graphs. This effectiveness was also evidenced on G-241N for the partitions of 100 labeled examples for RBF- and HM-based graphs. Since G-241C and G-241N lie near a 5-dimensional manifold (CHAPELLE; SCHÖLKOPF; ZIEN, 2006), we expect that higher order regularization with $p = 5$ would lead to better performances than other experimental settings.

On TEXT, the proposed method showed the best performances in the partitions of 10 labeled examples for all graph construction methods, excluding mutKNN-RBF. In addition, they showed the best performances in the partitions of 100 labeled examples for HM- and LLE-based graphs. In these partitions, CGFHF was outperformed by RMGT for RBF-based graphs.

Table 17 – Best average error rates (%) and standard deviations (%) for the SSL algorithms using symFKNN-LLE.

$l = 10$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	15.55 (2.74)	36.10 (5.87)	10.31 (4.68)	47.25 (4.14)	47.45 (4.36)	43.54 (6.95)	7.417
LGC	14.51 (2.81)	35.98 (6.06)	10.97 (5.00)	41.24 (4.37)	43.06 (3.34)	39.03 (3.82)	6.0
LapRLS	14.55 (3.37)	36.17 (4.81)	10.31 (4.63)	38.69 (5.56)	40.61 (5.52)	38.62 (4.13)	5.25
LapSVM	14.41 (3.23)	35.21 (4.58)	9.83 (3.99)	39.00 (6.40)	40.84 (6.40)	42.62 (4.92)	4.833
RMGT	18.96 (1.18)	30.41 (3.71)	7.73 (2.43)	42.75 (7.33)	41.77 (3.33)	36.25 (4.78)	5.5
CGFHF	14.22 (4.12)	28.93 (4.89)	6.15 (2.07)	41.82 (10.06)	36.37 (5.17)	30.60 (3.62)	3.083
CLGC	14.20 (4.11)	29.27 (3.99)	5.16 (2.32)	41.82 (10.08)	36.21 (4.95)	28.98 (3.21)	2.667
CLapRLS	12.40 (3.81)	29.19 (3.84)	4.29 (1.27)	36.58 (4.48)	36.21 (4.95)	28.50 (2.57)	1.25
$l = 100$	USPS	COIL ₂	DIGIT-1	G-241N	G-241C	TEXT	avg. ranking
GFHF	6.76 (2.30)	1.29 (1.57)	2.18 (0.44)	32.85 (4.42)	38.59 (4.66)	27.92 (3.82)	7.5
LGC	5.91 (0.79)	1.62 (1.81)	2.38 (0.85)	28.93 (1.80)	34.20 (2.63)	27.13 (2.68)	7.0
LapRLS	5.48 (1.01)	0.89 (0.71)	2.12 (0.47)	23.96 (2.17)	28.73 (3.44)	26.93 (2.60)	4.833
LapSVM	5.90 (1.41)	0.90 (0.94)	2.13 (0.60)	21.64 (3.80)	25.51 (5.58)	27.89 (3.66)	5.0
RMGT	8.96 (2.63)	0.85 (0.80)	1.99 (0.43)	28.47 (4.00)	31.92 (1.50)	23.17 (1.45)	5.333
CGFHF	3.75 (0.44)	0.61 (0.60)	1.95 (0.41)	21.77 (1.19)	28.39 (1.98)	22.11 (0.86)	3.333
CLGC	3.23 (0.51)	0.26 (0.37)	1.57 (0.46)	20.25 (1.45)	25.49 (1.07)	20.83 (1.14)	1.667
CLapRLS	3.24 (0.49)	0.18 (0.36)	1.56 (0.31)	18.87 (2.63)	21.70 (1.93)	20.84 (1.14)	1.333

Figure 25 – Average error rates for the SSL algorithms with respect to k using symKNN-RBF in the partitions of 10 labeled examples.

7.2 Evaluation of classifier stability

Fig. 25 shows the average error rates of the SSL algorithms with respect to k in the partitions of 10 labeled examples using symKNN-RBF, which is widely used in the SSL literature (ZHU, 2005; CHAPELLE; SCHÖLKOPF; ZIEN, 2006). On COIL₂, DIGIT-1, G-241C, and G-241N, the proposed methods achieved the best performances for most values of k . Interestingly,

the proposed methods achieved almost the same performances with respect to k on G-241C.

Figure 26 – Average error rates for the SSL algorithms with respect to k on G-241C and G-241N using mutKNN-based graphs in the partitions of 10 labeled examples.

On TEXT, the constrained methods achieved the four best performances for most values of k with similar performances for $k \geq 14$. On USPS, the proposed methods achieved competitive performances to the other SSL algorithms for all values of k .

In the machine learning community, it is not common to show negative results in the experimental analysis, probably due to the fear of rejection. However, to the best of our knowledge, achieving state-of-the-art performance with respect to graph construction on a variety of widely used benchmark data sets for all parameter settings is a hard (sometimes impossible) task.

We are not saying that someone is doing *cherry picking*³ in an attempt to increase the acceptance rate of the paper. We are just saying that reporting the negative results to the research community is a relevant approach in an attempt to identify weaknesses of the algorithms on specific scenarios and possibly providing insights on how to fix them on future research. Therefore, we show in the sequence the negative results related to the proposed constrained methods.

The proposed methods achieved state-of-the-art performance on USPS, COIL₂, DIGIT-1, and TEXT for *all* graph construction methods with respect to k . However, we found a few

³ Cherry picking is the act of pointing to individual cases that seem to confirm a particular hypothesis, while ignoring a significant portion of related cases that may contradict that hypothesis.

Figure 27 – Average error rates for the SSL algorithms with respect to k on G-241C and G-241N using mutKNN-based graphs in the partitions of 100 labeled examples.

experimental settings in which our methods showed good performance only for dense graphs (specially when $k \approx 40$).

In these settings, our methods were outperformed by LapRLS and LapSVM by a large margin on sparse graphs. However, our methods showed better performances than LapRLS and LapSVM for dense graphs. These negative results were evidenced only in the *artificial* data sets G-241N and G-241C for mutKNN-based graphs in both partitions.

We report the negative results in Table 18 and Fig. 26 and 27. Fig. 26 shows that CLapRLS achieved the best performances for all values of k on both data sets for mutKNN-LLE. For mutKNN-LLE, while CGFHF and CLGC showed no competitive performances with CLapRLS on G-241N, CGFHF and CLGC showed good performances only for $k \geq 38$ on G-241C.

Fig. 27 shows that CLapRLS achieved the best performances for all values of k on G-241C and on G-241N for mutKNN-LLE. However, CGFHF and CLGC showed competitive performances only for $k \approx 40$. On G-241N for mutKNN-RBF and mutKNN-HM, the proposed methods showed similar performances with respect to k , being competitive with LapRLS and LapSVM for $k \geq 20$.

Table 18 shows that the proposed methods achieved good to exceptional performances on most experimental settings. In addition, CLapRLS showed the best performances in most

Table 18 – Best average error rates (%) for the SSL algorithms in the experimental settings described in Fig. 26 and 27.

		GFHF	LGC	LapRLS	LapSVM	RMGT	CGFHF	CLGC	CLapRLS
G-241C (10)	mutKNN-RBF	46.72	41.85	40.66	40.90	42.77	33.26	32.92	32.30
	mutKNN-HM	46.50	41.94	40.67	40.90	40.53	34.12	34.03	33.89
	mutKNN-LLE	46.98	42.94	40.59	40.82	42.12	39.84	39.03	37.67
average ranking		8.0	6.667	4.333	5.333	5.667	3.0	2.0	1.0
		GFHF	LGC	LapRLS	LapSVM	RMGT	CGFHF	CLGC	CLapRLS
G-241N (10)	mutKNN-RBF	46.94	39.82	38.36	38.90	46.11	36.83	36.88	34.12
	mutKNN-HM	46.66	39.51	38.29	39.00	44.73	35.20	35.21	33.49
	mutKNN-LLE	47.14	42.12	38.66	38.95	43.75	41.38	41.38	36.61
average ranking		8.0	6.0	3.333	4.333	7.0	2.667	3.333	1.0
		GFHF	LGC	LapRLS	LapSVM	RMGT	CGFHF	CLGC	CLapRLS
G-241C (100)	mutKNN-RBF	38.62	34.45	28.73	25.51	33.12	26.48	24.59	21.70
	mutKNN-HM	38.96	34.70	28.73	25.51	32.02	27.29	25.65	21.70
	mutKNN-LLE	39.12	36.13	28.73	25.51	34.10	32.41	31.40	21.70
average ranking		8.0	7.0	4.333	2.333	6.0	4.333	3.0	1.0
		GFHF	LGC	LapRLS	LapSVM	RMGT	CGFHF	CLGC	CLapRLS
G-241N (100)	mutKNN-RBF	32.56	26.20	23.92	21.64	31.08	15.82	15.01	15.32
	mutKNN-HM	32.42	26.17	23.95	21.64	29.64	15.80	14.93	15.10
	mutKNN-LLE	33.78	30.22	23.96	21.64	29.93	23.67	21.85	18.91
average ranking		8.0	6.333	5.0	4.0	6.667	3.0	1.333	1.667

settings with the best average ranking in three out of four settings of data set and partition. On G-241N in the partitions of 100 labeled examples, CLapRLS showed equivalent performances to CLGC for mutKNN-RBF and mutKNN-HM. Therefore, one may conclude that CLapRLS would be the best suited SSL algorithm for G-241N and G-241C, independent of the graph construction method used.

Although the best case analysis is the most used empirical evaluation model in the SSL literature (CHAPELLE; SCHÖLKOPF; ZIEN, 2006; ZHU, 2005), Table 18 and Fig. 26 and 27 show that this analysis alone may not be adequate for a comprehensive experimental evaluation because it may hide useful information concerning the SSL algorithms' performance with respect to parameter selection. Without a comprehensive analysis on the SSL algorithms' performance with respect to graph construction and parameter selection, we can obtain tendentious conclusions on their performance, which is clearly undesired for the research community.

7.3 Evaluation of error surfaces

This section analyzes the error surfaces for CGFHF, CLGC, and CLapRLS with respect to their corresponding (regularization and external) parameters.

7.3.1 Error surfaces for CGFHF ($k \times p$)

Fig. 28 shows the error surfaces for CGFHF with respect to k and p using symKNN-RBF in the partitions of 10 labeled examples. The error surfaces for USPS and DIGIT-1 showed similar patterns, with the optimal performances occurring when $p = 2$. In these data sets, CGFHF

Figure 28 – Error surfaces for CGFHF with respect to k and p using symKNN-RBF in the partitions of 10 labeled examples.

showed strong dependency of parameter selection with respect to p . However, there is no abrupt changes in the performance of CGFHF with respect to k .

On COIL₂, the value of p showed a relatively small influence in the performance of CGFHF. However, the value of k showed a moderate influence in the performance of GFHF as p increases. On G-241C, CGFHF showed weak dependency of parameter selection with respect to k and p . The best performances were achieved for large values of k and p , specially for $p = 5$ and $k \geq 20$. On TEXT, CGFHF achieved its best performances for large values of k and p , specially for $p = 5$ and $k \approx 20$.

The error surfaces for CGFHF with respect to k and p using other graph construction methods were equivalent to those reported in Fig. 28. We found no other explorable pattern in the error surfaces.

7.3.2 Error surfaces for CLGC ($k \times p$)

Fig. 29 shows the error surfaces for CLGC with respect to k and p using symKNN-RBF in the partitions of 10 labeled examples. We obtained the same patterns shown in Fig. 28. Therefore, the experimental analysis remains the same as in Fig. 28.

The error surfaces for CLGC with respect to k and p using other graph construction methods were equivalent to those reported in Fig. 29. We found no other explorable pattern in

Figure 29 – Error surfaces for CLGC with respect to k and p using symKNN-RBF in the partitions of 10 labeled examples.

the error surfaces.

7.3.3 Error surfaces for CLGC ($k \times \mu$)

Fig. 30 shows the error surfaces for CLGC with respect to k and μ using symKNN-RBF in the partitions of 10 labeled examples. On DIGIT-1 and TEXT, the optimal performances occurred for large values of k and small values of μ . On USPS and G-241N, the optimal performances occurred for large values of k and μ . On COIL₂, CLGC showed weak dependency of parameter selection with respect to μ for $\mu \geq 0.05$ and a strong dependency with respect to k . On G-241C, CLGC showed weak dependency of parameter selection with respect to μ and the optimal performances occur for $k \geq 20$, for all μ .

Fig. 31 shows the error surfaces for CLGC with respect to k and μ on G-241C using RBF-based graphs in the partitions of 10 labeled examples. The error surfaces for symKNN-RBF and symFKNN-RBF showed similar patterns, with the best performances occurring for $k \geq 20$, for all μ . However, for mutKNN-RBF, the optimal performance occurs only for $k = 40$. The performances for $k \neq 40$ were unsatisfactory, independent of μ . The error surface for mutKNN-RBF shows that the parameter selection process for CLGC may be a hard task for some experimental settings.

Figure 30 – Error surfaces for CLGC with respect to k and μ using symKNN-RBF in the partitions of 10 labeled examples.

Figure 31 – Error surfaces for CLGC with respect to k and μ on G-241C using RBF-based graphs in the partitions of 10 labeled examples.

7.3.4 Error surfaces for CLGC ($p \times \mu$)

Fig. 32 shows the error surfaces for CLGC with respect to p and μ using symKNN-RBF in the partitions of 10 labeled examples. On G-241C, the value of μ showed low influence in the performance of CLGC while the value of p showed high influence. The optimal performances occurred for $p = 5$, independent of μ . On G-241N, the optimal performance also occurred for $p = 5$. On COIL₂, DIGIT-1, and USPS, the error surfaces showed similar patterns. The optimal performances occurred for $p = 2$, independent of μ . On TEXT, the optimal performances occurred for $p = 5$ and $\mu \leq 0.05$.

Figure 32 – Error surfaces for CLGC with respect to p and μ using symKNN-RBF in the partitions of 10 labeled examples.

The error surfaces for CLGC with respect to p and μ using other graph construction methods were equivalent to those reported in Fig. 32. We found no other explorable pattern in the error surfaces.

7.3.5 Error surfaces for CLapRLS ($k \times p$)

Fig. 33 shows the error surfaces for CLapRLS with respect to k and p using symKNN-RBF in the partitions of 10 labeled examples. The error surfaces for CLapRLS on COIL₂, DIGIT-1, G-241C, G-241N, and TEXT are similar to those reported by CGFHF and CLGC. On USPS, the optimal performances occurred for $p = 3$, independent of k .

The error surfaces for CLapRLS with respect to k and p using other graph construction methods were equivalent to those reported in Fig. 33. We found no other explorable pattern in the error surfaces.

7.3.6 Error surfaces for CLapRLS ($\gamma_A \times \gamma_I$)

Fig. 34 shows the error surfaces for CLapRLS with respect to γ_A and γ_I using symKNN-RBF in the partitions of 10 labeled examples. The error surfaces for CLapRLS on COIL₂, DIGIT-1, G-241C, TEXT, and USPS showed similar patterns. The optimal performances were achieved for large values of γ_I and low values of γ_A , which is a common parameter choice in

Figure 33 – Error surfaces for CLapRLS with respect to k and p using symKNN-RBF in the partitions of 10 labeled examples.

Figure 34 – Error surfaces for CLapRLS with respect to γ_A and γ_l using symKNN-RBF in the partitions of 10 labeled examples.

the SSL literature (BELKIN; NIYOGI; SINDHWANI, 2006; MELACCI; BELKIN, 2011). On G-241N, the error surface shows an opposite pattern to that for the other data sets.

Figure 35 – Error surfaces for CLapRLS with respect to γ_A and γ_I on G-241C using LLE-based graphs in the partitions of 10 labeled examples.

Most error surfaces for CLapRLS with respect to γ_A and γ_I using other graph construction methods were equivalent to those reported in Fig. 34. However, we found some results that could hinder the parameter selection process for CLapRLS with respect to γ_A and γ_I . This result is reported in Fig. 35, which shows the error surfaces for CLapRLS with respect to γ_A and γ_I on G-241C using LLE-based graphs in the partitions of 10 labeled examples. Specifically, we show that mutKNN-LLE shows an opposite pattern to that reported by symKNN-LLE and symFKNN-LLE, which error surfaces are equivalent to those for RBF- and HM-based graphs.

7.3.7 Error surfaces for CLapRLS ($k \times \gamma_A$)

Fig. 36 shows the error surfaces for CLapRLS with respect to k and γ_A using symKNN-RBF in the partitions of 10 labeled examples. The error surfaces for CLapRLS on COIL₂, DIGIT-1, G-241C, TEXT, and USPS showed similar patterns. The optimal performances occurred for low values of γ_A , which is a common parameter choice in the SSL literature (BELKIN; NIYOGI; SINDHWANI, 2006; MELACCI; BELKIN, 2011). However, on G-241N, CLapRLS showed poor performances for this parameter choice. On this data set, the optimal performances occurred for $\gamma_A = 10^{-2}$.

The error surfaces for CLapRLS with respect to k and γ_A using other graph construction methods were equivalent to those reported in Fig. 36. We found no other explorable pattern in the error surfaces.

7.3.8 Error surfaces for CLapRLS ($k \times \gamma_I$)

Fig. 37 shows the error surfaces for CLapRLS with respect to k and γ_I using symKNN-RBF in the partitions of 10 labeled examples. The error surfaces on COIL₂, DIGIT-1, G-241C, and TEXT showed similar patterns. The best performances were achieved for large values of γ_I .

Figure 36 – Error surfaces for CLapRLS with respect to k and γ_A using symKNN-RBF in the partitions of 10 labeled examples..

Figure 37 – Error surfaces for CLapRLS with respect to k and γ_l using symKNN-RBF in the partitions of 10 labeled examples.

However, this parameter choice lead to poor performance on G-241N. On this data set, the best performances were achieved for low values of γ .

The error surfaces for CLapRLS with respect to k and γ_I using other graph construction methods were equivalent to those reported in Fig. 37. We found no other explorable pattern in the error surfaces.

Figure 38 – Error surfaces for CLapRLS with respect to p and γ_A using symKNN-RBF in the partitions of 10 labeled examples.

7.3.9 Error surfaces for CLapRLS ($p \times \gamma_A$)

Fig. 38 shows the error surfaces for CLapRLS with respect to p and γ_A using symKNN-RBF in the partitions of 10 labeled examples. The error surfaces on COIL₂, DIGIT-1, G-241C, and TEXT showed that CLapRLS achieved the best performances for small values of γ_A with different values of p for each data set. On G-241N, the best performances were achieved for large values of γ_A , independent of p .

The error surfaces for CLapRLS with respect to p and γ_A using other graph construction methods were equivalent to those reported in Fig. 38. We found no other explorable pattern in the error surfaces.

Figure 39 – Error surfaces for CLapRLS with respect to p and γ_l using symKNN-RBF in the partitions of 10 labeled examples.

7.3.10 Error surfaces for CLapRLS ($p \times \gamma_l$)

Fig. 39 shows the error surfaces for CLapRLS with respect to p and γ_l using symKNN-RBF in the partitions of 10 labeled examples. On COIL₂ and DIGIT-1, the best performances were achieved for large values of γ_l and low values of p . On TEXT and USPS, the best performances were achieved for $\gamma_l = 10^2$ and $p = 4$. On G-241C, the best performances were achieved for $\gamma_l = 10^2$ and $p = 5$. However, on G-241N, large values of γ_l lead to poor performances.

The error surfaces for CLapRLS with respect to p and γ_l using other graph construction methods were equivalent to those reported in Fig. 39. We found no other explorable pattern in the error surfaces.

7.4 Our findings

This chapter showed our experimental analysis on constrained graph-based SSL. Our findings are summarized as follows:

1. CGFHF, CLGC, and CLapRLS achieved the best average rankings in both partitions with CLapRLS being the best ranked method in both partitions for most graph construction methods. Therefore, these methods are effective in SSL;

2. we reported in Fig. 26 and 27 some experimental settings in which the proposed methods showed good performances only for dense graphs, specially when $k \approx 40$. These negative results were evidenced only in the *artificial* data sets G-241N and G-241C for mutKNN-based graphs in both partitions;
3. since G-241C and G-241N lie near a 5-dimensional manifold (CHAPELLE; SCHÖLKOPF; ZIEN, 2006), we expect that higher order regularization with $p = 5$ would lead to better performances than other experimental settings. Therefore, we conjecture that the Laplacian's degree p is related to the number of dimensions of the manifold. This means that if the data lie in a r -dimensional manifold, we should set $p = r$ to achieve satisfactory performances, not necessarily the best ones;
4. we evaluated the error surfaces for CGFHF, CLGC, and CLapRLS. In most experimental settings for a given SSL algorithm and two parameters, we found that these error surfaces were equivalent, independent of the graph construction method used. This is an interesting result for practical applications because graph construction may have *low influence*⁴ in the performance of the proposed constrained methods.

⁴ Graph construction usually has high influence in the performance of unconstrained methods for graph-based SSL (SOUSA; REZENDE; BATISTA, 2013). The weak dependency of graph construction shown by our constrained methods is a great property to practitioners, who can obtain satisfactory performance choosing any graph construction method. Results on more data sets are required to verify the validity of this property.

CONCLUSION

In this thesis, we provided theoretical and experimental analyses on graph-based SSL. From our experimental results, we showed strengths and weaknesses of existing algorithms and how to overcome them, if applicable. We also proposed new constrained methods for graph-based SSL.

The main contributions in this thesis are summarized as follows:

1. we showed that the convergence proof in (ZHU; GHAHRAMANI, 2002; ZHU, 2005) can not be extended to sparse graphs. Specifically, we showed that there is a convergence restriction for that proof;
2. we proposed generalized iterative methods for GFHF and LGC under certain conditions;
3. we showed that LapRLS generalizes GFHF and LGC. Therefore, GFHF and LGC are naturally *inductive*¹ methods, instead of transductive ones;
4. we provided three new graph-based SSL algorithms applying the normalization constraints of Liu and Chang (2009) into the optimization problem of existing methods. We called the proposed methods CGFHF, CLGC, and CLapRLS;
5. we showed that CGFHF, CLGC, and CLapRLS have closed-form solution;
6. we proved that CLapRLS generalizes CLGC, CGFHF, and RMGT. This means that the proposed constrained methods, and also RMGT, are naturally inductive;
7. we showed that the kernel function has no influence in the transductive solution of CGFHF and CLGC;

¹ The original formulations of GFHF and LGC state that these methods are transductive (ZHU; GHAHRAMANI; LAFFERTY, 2003; ZHOU *et al.*, 2004).

8. we proposed optimized ways to compute CGFHF, CLGC, and CLapRLS under some conditions;
9. we generalized one of the constraints of [Liu and Chang \(2009\)](#) to *any* distribution. From this new constraint, we generalized CGFHF, CLGC, and CLapRLS to any distribution;
10. the use of the best case analysis alone may not be adequate for a comprehensive experimental evaluation on graph-based SSL. This analysis may hide useful information concerning the classifiers' performance with respect to graph construction and parameter selection;
11. we showed the effectiveness of CLapRLS, CLGC, and CGFHF on benchmark data sets over many graph construction methods and parameter settings;
12. we conjecture that the Laplacian's degree p is related to the number of dimensions of the manifold. This means that if the data lie in a r -dimensional manifold, we should set $p = r$ to achieve satisfactory performances, not necessarily the best ones;
13. we evaluated the error surfaces for CGFHF, CLGC, and CLapRLS. In most experimental settings for a given SSL algorithm and two parameters, we found that these error surfaces were equivalent, independent of the graph construction method used. This is an interesting result for practical applications because graph construction may have low influence in the performance of the proposed constrained methods.

BIBLIOGRAPHY

ALAMGIR, M.; LUXBURG, U. von. Shortest path distance in random k -nearest neighbor graphs. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2012. p. 1031–1038. Citation on page [95](#).

ARGYRIOU, A.; HERBSTER, M.; PONTIL, M. Combining graph laplacians for semi-supervised learning. In: **Advances in Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2006. Citation on page [100](#).

ARGYRIOU, A.; MICCHELLI, C. A.; PONTIL, M. Learning convex combinations of continuously parameterized basic kernels. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2005. Citation on page [100](#).

BAGNALL, A.; DAVIS, L.; HILLS, J.; LINES, J. Transformation based ensembles for time series classification. In: **SIAM International Conference on Data Mining (SDM)**. [S.l.: s.n.], 2012. Citation on page [79](#).

BATISTA, G. E. A. P. A.; KEOGH, E. J.; TATAW, O. M.; SOUZA, V. M. A. de. Cid: an efficient complexity-invariant distance for time series. **Data Mining and Knowledge Discovery**, v. 28, n. 3, p. 634–669, 2014. Citation on page [78](#).

BAYDOGAN, M.; RUNGER, G.; TUV, E. A bag-of-features framework to classify time series. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 35, n. 11, p. 2796–2802, 2013. Citation on page [78](#).

BELKIN, M.; NIYOGI, P.; SINDHWANI, V. Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. **Journal of Machine Learning Research**, v. 7, p. 2399–2434, 2006. Citations on pages [21](#), [22](#), [23](#), [24](#), [25](#), [27](#), [32](#), [54](#), [65](#), [83](#), [84](#), [91](#), [93](#), [94](#), [99](#), [117](#) e [137](#).

BEYER, K.; GOLDSTEIN, J.; RAMAKRISHNAN, R.; SHAFT, U. When is “nearest neighbor” meaningful? In: **ICDT**. [S.l.: s.n.], 1999. p. 217–235. Citation on page [68](#).

BOUSQUET, O.; ELISSEEFF, A. Stability and generalization. **Journal of Machine Learning Research**, v. 2, p. 499–526, 2002. Citation on page [24](#).

CHANG, H.; YEUNG, D.-Y. Robust locally linear embedding. **Pattern Recognition**, v. 39, n. 6, p. 1053–1065, 2006. Citations on pages [96](#) e [100](#).

CHAPELLE, O.; SCHÖLKOPF, B.; ZIEN, A. **Semi-supervised learning**. [S.l.]: The MIT Press, 2006. Citations on pages [21](#), [22](#), [23](#), [25](#), [67](#), [80](#), [84](#), [89](#), [107](#), [108](#), [111](#) e [121](#).

CHAPELLE, O.; ZIEN, A. Semi-supervised classification by low density separation. In: **Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)**. [S.l.: s.n.], 2005. p. 57–64. Citation on page [22](#).

CHUNG, F. **Spectral graph theory**. [S.l.]: American Mathematical Society, 1997. Citations on pages [31](#) e [141](#).

- CORTES, C.; MOHRI, M.; PECHYONY, D.; RASTOGI, A. Stability of transductive regression algorithms. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2008. p. 176–183. Citations on pages [23](#) e [24](#).
- COSTA, L. F.; RODRIGUES, F. A.; TRAVIESO, G.; BOAS, P. R. V. Characterization of complex networks: A survey of measurements. **Advances in Physics**, v. 56, n. 1, p. 167–242, 2007. Citation on page [99](#).
- DAI, G.; YEUNG, D. Kernel selection for semi-supervised kernel machines. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2007. p. 185–192. Citation on page [100](#).
- DAITCH, S. I.; KELNER, J. A.; SPIELMAN, D. A. Fitting a graph to vector data. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2009. p. 201–208. Citations on pages [22](#), [96](#) e [100](#).
- DEMME, J. W. **Applied numerical linear algebra**. [S.l.]: SIAM, 1997. Citations on pages [141](#) e [142](#).
- DEMsAR, J. Statistical comparisons of classifiers over multiple data sets. **Journal of Machine Learning Research**, v. 7, p. 1–30, 2006. Citation on page [81](#).
- DENG, H.; RUNGER, G.; TUV, E.; VLADIMIR, M. A time series forest for classification and feature extraction. **Information Science**, v. 239, p. 142–153, 2013. Citation on page [78](#).
- EL-YANIV, R.; PECHYONY, D. Stable transductive learning. In: **Proceedings of the Conference on Learning Theory (COLT)**. [S.l.: s.n.], 2006. p. 35–49. Citation on page [24](#).
- FANG, Y.; CHANG, K.; LAUW, H. Graph-based semi-supervised learning: Realizing pointwise smoothness probabilistically. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2014. p. 406–414. Citation on page [24](#).
- FRIEDMAN, M. A comparison of alternative tests of significance for the problem of m rankings. **Annals of Mathematical Statistics**, v. 11, p. 86–92, 1940. Citations on pages [81](#) e [104](#).
- GENG, B.; XU, C.; TAO, D.; YANG, L.; HUA, X.-S. Ensemble manifold regularization. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2009. p. 2396–2402. Citations on pages [83](#) e [84](#).
- GHAZVININEJAD, M.; MAHDIEH, M.; RABIEE, H. R.; ROSHAN, P. K.; ROHBAN, M. H. Isograph: Neighbourhood graph construction based on geodesic distance for semi-supervised learning. In: **Proceedings of the International Conference on Data Mining (ICDM)**. [S.l.: s.n.], 2011. p. 191–200. Citation on page [22](#).
- GOLUB, G. H.; LOAN, C. F. van. **Matrix computations**. [S.l.]: Johns Hopkins University Press, 1996. Citations on pages [141](#) e [142](#).
- GRANDVALET, Y.; BENGIO, Y. Semi-supervised learning by entropy minimization. In: **Advances in Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2005. p. 529–536. Citation on page [22](#).
- HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction**. 2. ed. [S.l.]: Springer, 2009. Citation on page [135](#).

HE, H.; GARCIA, E. A. Learning from imbalanced data. **IEEE Transactions on Knowledge and Data Engineering**, v. 21, n. 9, p. 1263–1284, 2000. Citation on page 98.

HEIN, M.; AUDIBERT, J.-Y.; LUXBURG, U. von. Graph laplacians and their convergence on random neighborhood graphs. **Journal of Machine Learning Research**, v. 8, p. 1325–1368, 2007. Citation on page 28.

HEIN, M.; MAIER, M. Manifold denoising. In: **Advances in Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2007. p. 561–568. Citations on pages 22 e 70.

HILLS, J.; LINES, J.; BARANAUSKAS, E.; MAPP, J.; BAGNALL, A. Classification of time series by shapelet transformation. **Data Mining and Knowledge Discovery**, 2013. Citation on page 78.

HUANG, B.; JEBARA, T. Loopy belief propagation for bipartite maximum weight b-matching. In: **Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)**. [S.l.: s.n.], 2007. p. 195–202. Citation on page 22.

_____. Fast b-matching via sufficient selection belief propagation. In: **Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)**. [S.l.: s.n.], 2011. p. 361–369. Citation on page 22.

JEBARA, T.; WANG, J.; CHANG, S.-F. Graph construction and b-matching for semi-supervised learning. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2009. p. 441–448. Citations on pages 69, 70, 89 e 99.

JOACHIMS, T. Transductive learning via spectral graph partitioning. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2003. p. 290–297. Citation on page 22.

JOHNSON, R.; ZHANG, T. On the effectiveness of laplacian normalization for graph semi-supervised learning. **Journal of Machine Learning Research**, v. 8, p. 1489–1517, 2007. Citations on pages 72, 153 e 155.

KAPOOR, A.; QI, Y.; AHN, H.; PICARD, R. Hyperparameter and kernel learning for graph based semi-supervised classification. In: **Advances in Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2005. p. 1585–1592. Citation on page 98.

KEOGH, E. Exact indexing of dynamic time warping. In: **Proceedings of the International Conference on Very Large Databases (VLDB)**. [S.l.: s.n.], 2002. p. 406–417. Citation on page 69.

KONG, D.; DING, C.; HUANG, H.; NIE, F. An iterative locally linear embedding algorithm. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2012. p. 1647–1654. Citations on pages 96 e 100.

KRISHNAPURAM, B.; WILLIAMS, D.; XUE, Y.; HARTEMINK, A.; CARIN, L.; FIGUEIREDO, M. On semi-supervised classification. In: **Advances in Neural Information Processing Systems (NIPS)** 17. [S.l.: s.n.], 2005. Citation on page 22.

LAWRENCE, N. D.; JORDAN, M. I. Semi-supervised learning via gaussian processes. In: **Advances in Neural Information Processing Systems (NIPS)** 17. [S.l.: s.n.], 2005. Citation on page 22.

LI, X.; LIN, S.; YAN, S.; XU, D. Discriminant locally linear embedding with high-order tensor data. **IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics**, v. 38, n. 2, p. 342–352, 2008. Citations on pages [96](#) e [100](#).

LIN, J.; KHADE, R.; LI, Y. Rotation-invariant similarity in time series using bag-of-patterns representation. **Journal of Intelligent Information Systems**, v. 39, n. 2, p. 287–315, 2012. Citation on page [78](#).

LINES, J.; BAGNALL, A. Time series classification with ensembles of elastic distance measures. **Data Mining and Knowledge Discovery**, 2014. Citations on pages [78](#) e [79](#).

LIU, H.; YANG, Y. Bipartite edge prediction via transductive learning over product graphs. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2015. p. 1880–1888. Citation on page [24](#).

LIU, T.; MOORE, A. W.; GRAY, A.; YANG, K. An investigation of practical approximate nearest neighbor algorithms. In: **Advances in Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2005. p. 825–832. Citation on page [95](#).

LIU, W.; CHANG, S.-F. Robust multi-class transductive learning with graphs. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2009. p. 381–388. Citations on pages [24](#), [25](#), [27](#), [33](#), [34](#), [43](#), [44](#), [48](#), [54](#), [65](#), [66](#), [70](#), [73](#), [123](#), [124](#) e [145](#).

LIU, W.; HE, J.; CHANG, S.-F. Large graph construction for scalable semi-supervised learning. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2010. p. 679–686. Citations on pages [21](#), [22](#), [71](#) e [96](#).

LIU, W.; QIAN, B.; CUI, J.; LIU, J. Spectral kernel learning for semi-supervised classification. In: **Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)**. [S.l.: s.n.], 2009. p. 1150–1155. Citations on pages [21](#) e [22](#).

LUO, D.; DING, C.; HUANG, H.; NIE, F. Forging the graphs: A low rank and positive semidefinite graph learning approach. In: **Advances in Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2012. p. 2969–2977. Citation on page [22](#).

LUXBURG, U. von. A tutorial on spectral clustering. **Statistics and Computing**, v. 17, n. 4, p. 395–416, 2007. Citation on page [95](#).

LUXBURG, U. von; ALAMGIR, M. Density estimation from unweighted k-nearest neighbor graphs: a roadmap. In: **Advances in Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2013. p. 225–233. Citation on page [95](#).

MAHDAVIANI, M.; FREITAS, O. D.; FRASER, B.; HAMZE, F. Fast computational methods for visually guided robots. In: **ICRA**. [S.l.: s.n.], 2005. Citation on page [141](#).

MAIER, M.; HEIN, M.; LUXBURG, U. von. Cluster identification in nearest-neighbor graphs. In: **Proceedings of the Conference on Learning Theory (COLT)**. [S.l.: s.n.], 2007. p. 196–210. Citation on page [95](#).

_____. Optimal construction of k -nearest neighbor graphs for identifying noisy clusters. **Theoretical Computer Science**, v. 410, n. 19, p. 1749–1764, 2009. Citation on page [95](#).

- MAIER, M.; LUXBURG, U. von; HEIN, M. Influence of graph construction on graph-based clustering measures. In: **Advances in Neural Information Processing Systems (NIPS)** **21**. [S.l.: s.n.], 2009. p. 1025–1032. Citation on page [95](#).
- MANN, G. S.; MCCALLUM, A. Simple, robust, scalable semi-supervised learning via expectation regularization. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2007. p. 593–600. Citation on page [22](#).
- MELACCI, S.; BELKIN, M. Laplacian support vector machines trained in the primal. **Journal of Machine Learning Research**, v. 12, p. 1149–1184, 2011. Citations on pages [72](#), [91](#), [93](#), [94](#) e [117](#).
- MICCHELLI, C.; PONTIL, M. Learning the kernel function via regularization. **Journal of Machine Learning Research**, v. 6, p. 1099–1125, 2005. Citation on page [100](#).
- MUEEN, A.; KEOGH, E. Extracting optimal performance from dynamic time warping. In: **Proceedings of the International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD)**. [S.l.: s.n.], 2016. p. 2129–2130. Citation on page [69](#).
- NEMENYI, P. B. **Distribution-free multiple comparisons**. Phd Thesis (PhD Thesis) — Princeton University, 1963. Citations on pages [81](#) e [104](#).
- NIE, F.; WANG, H.; HUANG, H.; DING, C. Unsupervised and semi-supervised learning via l_1 -norm graph. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2011. p. 2268–2273. Citation on page [22](#).
- OZAKI, K.; SHIMBO, M.; KOMACHI, M.; MATSUMOTO, Y. Using the mutual k-nearest neighbor graphs for semi-supervised classification of natural language data. In: **CoNLL**. [S.l.: s.n.], 2011. p. 154–162. Citation on page [70](#).
- RAKTHANMANON, T.; KEOGH, E. Fast-shapelets: A fast algorithm for discovering robust time series shapelets. In: **SIAM International Conference on Data Mining (SDM)**. [S.l.: s.n.], 2013. Citation on page [79](#).
- ROSSET, S.; ZHU, J.; ZHOU, H.; HASTIE, T. A method for inferring label sampling mechanisms in semi-supervised learning. In: **Advances in Neural Information Processing Systems (NIPS)** **15**. [S.l.: s.n.], 2005. Citation on page [22](#).
- ROWEIS, S.; SAUL, L. Nonlinear dimensionality reduction by locally linear embedding. **Science**, v. 290, n. 5500, p. 2323–2326, 2000. Citations on pages [22](#) e [70](#).
- SHIVANNA, R.; CHATTERJEE, B. K.; SANKARAN, R.; BHATTACHARYYA, C.; BACH, F. Spectral norm regularization of orthonormal representations for graph transduction. In: **Proceedings of the International Conference on Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2015. p. 2215–2223. Citation on page [24](#).
- SILVA, D. F.; SOUZA, V. M. A. de; BATISTA, G. E. A. P. A. Time series classification using compression distance of recurrence plots. In: **IEEE International Conference on Data Mining (ICDM)**. [S.l.: s.n.], 2013. p. 687–696. Citation on page [78](#).
- SINDHWANI, V.; KEERTHI, S. S. Large scale semi-supervised linear SVMs. In: **ACM SIGIR**. [S.l.: s.n.], 2006. p. 477–484. Citation on page [22](#).

SINDHWANI, V.; NIYOGI, P.; BELKIN, M. Beyond the point cloud: from transductive to semi-supervised learning. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2005. p. 824–831. Citations on pages [83](#), [84](#) e [99](#).

SINDHWANI, V.; ROSENBERG, D. S. An rkhs for multi-view learning and manifold co-regularization. In: **Proceedings of the IEEE International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2008. p. 976–983. Citations on pages [83](#) e [84](#).

SOLOMON, J.; RUSTAMOV, R.; GUIBAS, L.; BUTSCHER, A. Wasserstein propagation for semi-supervised learning. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2014. p. 306–314. Citation on page [24](#).

SOUSA, C. A. R. An overview on the gaussian fields and harmonic functions method for semi-supervised learning. In: **Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)**. [S.l.: s.n.], 2015. p. 1439–1446. Citation on page [134](#).

SOUSA, C. A. R.; BATISTA, G. E. A. P. A. Robust multi-class graph transduction with higher order regularization. In: **Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)**. [S.l.: s.n.], 2015. p. 299–306. Citations on pages [43](#) e [103](#).

_____. Constrained local and global consistency for semi-supervised learning. In: **Proceedings of the IEEE International Conference on Pattern Recognition (ICPR)**. [S.l.: s.n.], 2016. Citations on pages [43](#) e [103](#).

SOUSA, C. A. R.; REZENDE, S. O.; BATISTA, G. E. A. P. A. Influence of graph construction on semi-supervised learning. In: **Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery on Databases (ECML/PKDD)**. [S.l.: s.n.], 2013. p. 160–175. Citations on pages [22](#), [23](#), [27](#), [36](#), [67](#), [68](#), [70](#), [71](#), [72](#), [73](#), [75](#), [76](#), [87](#), [90](#), [101](#), [103](#) e [121](#).

SOUSA, C. A. R.; SOUZA, V. M. A.; BATISTA, G. E. A. P. A. Time series transductive classification on imbalanced data sets: an experimental study. In: **Proceedings of the International Conference on Pattern Recognition (ICPR)**. [S.l.: s.n.], 2014. p. 3780–3785. Citations on pages [22](#) e [77](#).

_____. An experimental analysis on time series transductive classification on graphs. In: **Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)**. [S.l.: s.n.], 2015. p. 307–314. Citations on pages [22](#) e [77](#).

SPIELMAN, D. A.; TENG, S.-H. Nearly-linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. **ArXiv:cs/0607105v4**, 2006. Citations on pages [150](#) e [154](#).

SUBRAMANYA, A.; BILMES, J. Semi-supervised learning with measure propagation. **Journal of Machine Learning Research**, v. 12, p. 3311–3370, 2011. Citation on page [24](#).

SZUMMER, M.; JAAKKOLA, T. Partially labeled classification with markov random walks. In: **Advances in Neural Information Processing Systems (NIPS) 14**. [S.l.: s.n.], 2001. Citation on page [22](#).

_____. Information regularization with partially labeled data. In: **Advances in Neural Information Processing Systems (NIPS) 15**. [S.l.: s.n.], 2002. Citation on page [22](#).

TONG, W.; JIN, R. Semi-supervised learning by mixed label propagation. In: **AAAI**. [S.l.: s.n.], 2007. p. 651–656. Citation on page 22.

TSANG, I. W.; KWOK, J. T. Large-scale sparsified manifold regularization. In: **Advances in Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2007. p. 1401–1408. Citations on pages 83 e 84.

URNER, R.; SHALEV-SHWARTZ, S.; BEN-DAVID, S. Access to unlabeled data can speed up prediction time. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2011. p. 641–648. Citation on page 21.

WANG, J.; CHANG, S.; ZHOU, X.; WONG, S. Active microscopic cellular image annotation by superposable graph transduction with imbalanced labels. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2008. p. 1–8. Citations on pages 97 e 98.

WANG, J.; XIA, Y. Fast graph construction using auction algorithm. In: **Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)**. [S.l.: s.n.], 2012. p. 873–882. Citation on page 22.

WU, M.; Schölkopf, B. Transductive classification via local learning regularization. In: **Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)**. [S.l.: s.n.], 2007. p. 628–635. Citation on page 24.

WU, X.-M.; LI, Z.; SO, A. M.-C.; WRIGHT, J.; CHANG, S.-F. Learning with partially absorbing random walks. In: **Advances in Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2012. p. 3086–3094. Citation on page 22.

XIANG, S.; NIE, F.; PAN, C.; ZHANG, C. Regression reformulations of LLE and LTSA with locally linear transformation. **IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics**, v. 41, n. 5, p. 1250–1262, 2011. Citations on pages 96 e 100.

YANG, X.; BAI, X.; LATECKI, L. J.; TU, Z. Improving shape retrieval by learning graph transduction. In: **Proceedings of the European Conference on Computer Vision (ECCV)**. [S.l.: s.n.], 2008. p. 788–801. Citations on pages 97 e 98.

YANG, Z.; COHEN, W.; SALAKHUDINOV, R. Revisiting semi-supervised learning with graph embeddings. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2016. p. 40–48. Citation on page 24.

YE, L.; KEOGH, E. Time series shapelets: A new primitive for data mining. In: **ACM International Conference on Knowledge Discovery and Data Mining (KDD)**. [S.l.: s.n.], 2009. Citation on page 79.

ZASS, R.; SHASHUA, A. Doubly stochastic normalization for spectral clustering. In: **Advances in Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2006. p. 1569–1576. Citation on page 22.

ZELNIK-MANOR; PERONA, P. Self-tuning spectral clustering. In: **Advances in Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2005. p. 1601–1608. Citations on pages 97 e 98.

ZHANG, X.; LEE, W. Hyperparameter learning for graph based semi-supervised learning algorithms. In: **Advances in Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2007. p. 1585–1592. Citation on page [98](#).

ZHANG, Y.; ZHOU, Z.-H. Non-metric label propagation. In: **Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)**. [S.l.: s.n.], 2009. p. 1357–1362. Citation on page [22](#).

ZHANG, Y.-M.; HUANG, K.; LIU, C.-L. Fast and robust graph-based transductive learning via minimum tree cut. In: **Proceedings of the IEEE International Conference on Data Mining (ICDM)**. [S.l.: s.n.], 2011. p. 952–961. Citation on page [22](#).

ZHANG, Z.; WANG, J. MLLE: Modified locally linear embedding using multiple weights. In: **Advances in Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2007. p. 1593–1600. Citations on pages [96](#) e [100](#).

ZHAO, D. Formulating LLE using alignment technique. **Pattern Recognition**, v. 39, n. 11, p. 2233–2235, 2006. Citations on pages [96](#) e [100](#).

ZHOU, D.; BOUSQUET, O.; LAL, T.; WESTON, J.; Schölkopf, B. Learning with local and global consistency. In: **Advances in Neural Information Processing Systems (NIPS) 16**. [S.l.: s.n.], 2004. p. 321–328. Citations on pages [21](#), [22](#), [23](#), [24](#), [25](#), [27](#), [30](#), [31](#), [38](#), [41](#), [48](#), [65](#), [89](#), [123](#) e [151](#).

ZHOU, Z.-H.; LI, M. Tri-training: exploiting unlabeled data using three classifiers. **IEEE Transactions on Knowledge and Data Engineering**, v. 17, p. 1529–1541, 2005. Citation on page [22](#).

ZHU, X. **Semi-Supervised Learning Literature Survey**. [S.l.], 2005. Citations on pages [21](#), [22](#), [23](#), [36](#), [40](#), [41](#), [80](#), [84](#), [89](#), [108](#), [111](#), [123](#) e [153](#).

ZHU, X.; GHAMRANI, Z. **Learning from labeled and unlabeled data with label propagation**. Pittsburgh, 2002. Citations on pages [29](#), [39](#), [40](#), [41](#) e [123](#).

ZHU, X.; GHAMRANI, Z.; LAFFERTY, J. Semi-supervised learning using gaussian fields and harmonic functions. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2003. p. 912–919. Citations on pages [21](#), [22](#), [23](#), [24](#), [25](#), [27](#), [29](#), [41](#), [44](#), [65](#), [73](#), [98](#) e [123](#).

ZHU, X.; KANDOLA, J.; GHAMRANI, Z.; LAFFERTY, J. Nonparametric transforms of graph kernels for semi-supervised learning. In: **Advances in Neural Information Processing Systems (NIPS)**. [S.l.: s.n.], 2005. p. 1641–1648. Citation on page [100](#).

ZHU, X.; LAFFERTY, J. Harmonic mixtures: combining mixture models and graph based methods for inductive and scalable semi-supervised learning. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.: s.n.], 2005. p. 1052–1059. Citation on page [22](#).

ZHUANG, L.; GAO, H.; LIN, Z.; MA, Y.; ZHANG, X.; YU, N. Non-negative low rank and sparse graph for semi-supervised learning. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2012. p. 2328–2335. Citation on page [22](#).

HEURISTIC INDUCTIVE SOLUTION FOR GFHF

This chapter proposes a heuristic inductive solution for GFHF. After training stage, we consider that all examples in $\mathcal{X}$ are labeled. Specifically, each $\mathbf{x}_i \in \mathcal{X}$ has a corresponding “label vector”¹ $\mathbf{F}_{i,\cdot} \in \mathbb{R}^c$, which comes from the transductive solution $\mathbf{F}$. Let $\tilde{\mathcal{X}} := \mathcal{X} \cup \{\tilde{\mathbf{x}}\} \subset \mathbb{R}^d$ be a sample in the input space where $\tilde{\mathbf{x}} \in \mathbb{R}^d$ is a test example we want to classify. The main idea of our heuristic inductive solution is to find the transductive solution of GFHF using $\tilde{\mathcal{X}}$ and provide an approximated solution for $\tilde{\mathbf{x}}$ as fast as possible. We call our approach *Heuristic GFHF* (H-GFHF).

Let $\tilde{\mathbf{W}} \in \mathbb{R}^{(n+1) \times (n+1)}$ be the weighted matrix associated to $\tilde{\mathcal{X}}$. Since self-loops are not permitted, $\tilde{\mathbf{W}}_{\mathcal{U}\mathcal{U}} = 0$. Considering $\tilde{\mathcal{N}} \subset \mathcal{X}$ the set of neighbors of $\tilde{\mathbf{x}}$, we have $(\tilde{\mathbf{W}}_{\mathcal{U}\mathcal{L}}^\top)_i = \mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_i)$ if $\mathbf{x}_i \in \tilde{\mathcal{N}}$ and 0 otherwise. In order to decrease the computation time required for induction, we consider that Assumption A.1 holds². Mathematically, such an assumption states that $\tilde{\mathbf{W}}_{\mathcal{L}\mathcal{L}} = \mathbf{W}$.

Assumption A.1 (*Graph connectivity for induction*). *The graph connectivity on the training examples does not change during the induction process. Intuitively, we assume that $\mathbf{W}$ is constant.*

Let $\tilde{\mathbf{N}} \in \mathbb{B}^n$ be a vector in which $\tilde{\mathbf{N}}_i = 1$ if and only if $\mathbf{x}_i \in \tilde{\mathcal{N}}$, $\Psi : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}_+$ a distance function, and $\tilde{\Psi} \in \mathbb{R}^n$ a distance vector in which $\tilde{\Psi}_i := \Psi(\tilde{\mathbf{x}}, \mathbf{x}_i)$, $\forall i \in \mathbb{N}_n$. Consider $\tilde{\mathbf{F}} \in \mathbb{R}^{(n+1) \times c}$ the output matrix of GFHF for the “extended” training sample $\tilde{\mathcal{X}}$ and $\tilde{\mathbf{Y}}_{\mathcal{L}} := \mathbf{F} \in \mathbb{R}^{n \times c}$ the corresponding label matrix.

¹ Differently from the training stage, now we have *soft* (real-valued) labels, instead of binary ones.

² If Assumption A.1 does not hold, we have to reconstruct $\mathbf{W}$ for each test example $\tilde{\mathbf{x}}$. This is ineffective even for a small training sample.

For induction, the optimization problem in (2.3) becomes:

$$\min_{\tilde{\mathbf{F}} \in \mathbb{R}^{(n+1) \times c}} \text{tr}(\tilde{\mathbf{F}}^\top \tilde{\mathbf{L}} \tilde{\mathbf{F}}) \quad \text{s.t.} \quad \tilde{\mathbf{F}}_{\mathcal{L}} = \mathbf{F} \quad (\text{A.1})$$

$\tilde{\mathbf{L}} \in \mathbb{R}^{(n+1) \times (n+1)}$ is the graph Laplacian generated from $\widetilde{\mathbf{W}}$. This yields the following closed-form solution:

$$\tilde{\mathbf{F}} = \begin{bmatrix} \mathbf{F} \\ -\tilde{\mathbf{L}}_{\mathcal{U}\mathcal{U}}^{-1} \tilde{\mathbf{L}}_{\mathcal{U}\mathcal{L}} \mathbf{F} \end{bmatrix} \quad (\text{A.2})$$

Therefore, we classify $\tilde{\mathbf{x}}$ as:

$$y(\tilde{\mathbf{x}}) = \arg \max_{i \in \mathbb{N}_c} \left(\tilde{\mathbf{F}}_{\mathcal{U}} \right)_i = \arg \max_{i \in \mathbb{N}_c} \left(-\tilde{\mathbf{L}}_{\mathcal{U}\mathcal{U}}^{-1} \tilde{\mathbf{L}}_{\mathcal{U}\mathcal{L}} \mathbf{F} \right)_i \quad (\text{A.3})$$

Eq. (A.3) provides an inductive solution for GFHF for a given graph Laplacian $\tilde{\mathbf{L}}$. In Propositions A.2 and A.3, we provide special cases of (A.3) for $\tilde{\mathbf{L}} = \tilde{\mathbf{L}}_{\mathbb{C}}$ and $\tilde{\mathbf{L}} = \tilde{\mathbf{L}}_{\mathbb{N}}$, respectively.

Proposition A.2 ((SOUSA, 2015)). *If $\tilde{\mathbf{L}} = \tilde{\mathbf{L}}_{\mathbb{C}}$, (A.3) yields:*

$$y(\tilde{\mathbf{x}}) = \arg \max_{i \in \mathbb{N}_c} \sum_{\mathbf{x}_j \in \tilde{\mathcal{N}}} \mathcal{H}(\tilde{\mathbf{x}}, \mathbf{x}_j) \mathbf{F}_{ji} \quad (\text{A.4})$$

Proof. By construction and Assumption A.1, we have:

$$\left(\tilde{\mathbf{L}}_{\mathbb{C}} \right)_{\mathcal{U}\mathcal{U}} = \sum_{\mathbf{x}_k \in \tilde{\mathcal{N}}} \mathcal{H}(\tilde{\mathbf{x}}, \mathbf{x}_k), \quad \left(\left(\tilde{\mathbf{L}}_{\mathbb{C}} \right)_{\mathcal{U}\mathcal{L}} \right)_j = -\mathcal{H}(\tilde{\mathbf{x}}, \mathbf{x}_j) \tilde{\mathbf{N}}_j$$

Therefore, (A.3) yields:

$$\begin{aligned} y(\tilde{\mathbf{x}}) &= \arg \max_{i \in \mathbb{N}_c} \left(-\left(\tilde{\mathbf{L}}_{\mathbb{C}} \right)_{\mathcal{U}\mathcal{U}}^{-1} \left(\tilde{\mathbf{L}}_{\mathbb{C}} \right)_{\mathcal{U}\mathcal{L}} \mathbf{F} \right)_i = \arg \max_{i \in \mathbb{N}_c} \frac{\sum_{\mathbf{x}_j \in \tilde{\mathcal{N}}} \mathcal{H}(\tilde{\mathbf{x}}, \mathbf{x}_j) \mathbf{F}_{ji}}{\sum_{\mathbf{x}_k \in \tilde{\mathcal{N}}} \mathcal{H}(\tilde{\mathbf{x}}, \mathbf{x}_k)} \\ &= \arg \max_{i \in \mathbb{N}_c} \sum_{\mathbf{x}_j \in \tilde{\mathcal{N}}} \mathcal{H}(\tilde{\mathbf{x}}, \mathbf{x}_j) \mathbf{F}_{ji} \end{aligned}$$

□

Proposition A.3. *If $\tilde{\mathbf{L}} = \tilde{\mathbf{L}}_{\mathbb{N}}$, (A.3) yields:*

$$y(\tilde{\mathbf{x}}) = \arg \max_{i \in \mathbb{N}_c} \sum_{\mathbf{x}_j \in \tilde{\mathcal{N}}} \frac{\mathcal{H}(\tilde{\mathbf{x}}, \mathbf{x}_j) \mathbf{F}_{ji}}{\sqrt{\mathbf{D}_{jj} + \mathcal{H}(\tilde{\mathbf{x}}, \mathbf{x}_j)}} \quad (\text{A.5})$$

Proof. By construction, $\left(\tilde{\mathbf{L}}_{\mathbb{N}}\right)_{\mathcal{U}\mathcal{U}} = 1$. From Assumption A.1, we have:

$$\begin{aligned} \left(\left(\tilde{\mathbf{L}}_{\mathbb{N}}\right)_{\mathcal{U}\mathcal{L}}\right)_j &= -\frac{\left(\tilde{\mathbf{W}}_{\mathcal{U}\mathcal{L}}\right)_j}{\sqrt{\tilde{\mathbf{D}}_{\mathcal{U}\mathcal{U}}}\sqrt{\left(\tilde{\mathbf{D}}_{\mathcal{L}\mathcal{L}}\right)_{jj}}} \\ &= -\frac{\mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_j)\tilde{\mathbf{N}}_j}{\sqrt{\sum_{\mathbf{x}_k \in \mathcal{N}} \mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_k)}\sqrt{\mathbf{D}_{jj} + \mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_j)\tilde{\mathbf{N}}_j}} \end{aligned} \quad (\text{A.6})$$

Therefore, (A.3) yields:

$$\begin{aligned} y(\tilde{\mathbf{x}}) &= \arg \max_{i \in \mathbb{N}_c} \left(-\left(\tilde{\mathbf{L}}_{\mathbb{N}}\right)_{\mathcal{U}\mathcal{U}}^{-1} \left(\tilde{\mathbf{L}}_{\mathbb{N}}\right)_{\mathcal{U}\mathcal{L}} \mathbf{F} \right)_i = \arg \max_{i \in \mathbb{N}_c} - \sum_{\mathbf{x}_j \in \mathcal{N}} \left(\left(\tilde{\mathbf{L}}_{\mathbb{N}}\right)_{\mathcal{U}\mathcal{L}}\right)_j \mathbf{F}_{ji} \\ &= \arg \max_{i \in \mathbb{N}_c} \sum_{\mathbf{x}_j \in \mathcal{N}} \frac{\mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_j) \mathbf{F}_{ji}}{\sqrt{\mathbf{D}_{jj} + \mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_j)}} \end{aligned} \quad (\text{A.7})$$

□

Proposition A.2 shows that (A.3) is equivalent to *Nadaraya-Watson kernel regression* (HASTIE; TIBSHIRANI; FRIEDMAN, 2009) for multi-class problems when $\tilde{\mathbf{L}} = \tilde{\mathbf{L}}_{\mathbb{C}}$. Given a kernel function $K_h(\cdot)$ with bandwidth $h > 0$, a sample $\{\mathbf{x}_i\}_{i=1}^n \subset \mathbb{R}^d$, and target values $\{y_i\}_{i=1}^n \subset \mathbb{R}$, the Nadaraya-Watson kernel regression provides an estimator $\hat{m}_h(\mathbf{x})$ such that:

$$\hat{m}_h(\mathbf{x}) = \frac{\sum_{i=1}^n K_h(\mathbf{x} - \mathbf{x}_i) y_i}{\sum_{i=1}^n K_h(\mathbf{x} - \mathbf{x}_i)}$$

Proposition A.3 provides a similar classification function than in Proposition A.2, incorporating the additional normalization factor $\sqrt{\mathbf{D}_{jj} + \mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_j)}$. Therefore, we classify $\tilde{\mathbf{x}}$ as:

$$y(\tilde{\mathbf{x}}) = \arg \max_{i \in \mathbb{N}_c} \sum_{\mathbf{x}_j \in \mathcal{N}} \Gamma_{\tilde{\mathbf{L}}}(\tilde{\mathbf{x}}, \mathbf{x}_j) \mathbf{F}_{ji}$$

such that

$$\Gamma_{\tilde{\mathbf{L}}}(\tilde{\mathbf{x}}, \mathbf{x}_j) = \begin{cases} \mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_j), & \tilde{\mathbf{L}} = \tilde{\mathbf{L}}_{\mathbb{C}} \\ \mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_j) (\mathbf{D}_{jj} + \mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_j))^{-1/2}, & \tilde{\mathbf{L}} = \tilde{\mathbf{L}}_{\mathbb{N}} \end{cases}$$

KERNEL APPROACH FOR GFHF

This chapter formulates our kernel approach for GFHF, called *Kernel GFHF* (K-GFHF). Assume that $(\mathbf{A}^{-1})_{\bullet\circ}$ is the submatrix $\bullet\circ$ of $\mathbf{A}^{-1}$ while $\mathbf{A}_{\bullet\bullet}^{-1}$ is the inverse of $\mathbf{A}_{\bullet\bullet}$, such that $\bullet$ and $\circ$ can be either $\mathcal{L}$ or $\mathcal{U}$. We start our theoretical analysis with Proposition B.1, which is a well known block matrix inversion property.

Proposition B.1 (*Properties of $\mathbf{K}$*). *Since $\mathbf{K}$ is symmetric and invertible, we have:*

$$\begin{aligned}
 (\mathbf{K}^{-1})_{\mathcal{L}\mathcal{L}} &= \left(\mathbf{K}_{\mathcal{L}\mathcal{L}} - \mathbf{K}_{\mathcal{U}\mathcal{L}}^{\top} \mathbf{K}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}} \right)^{-1} \\
 (\mathbf{K}^{-1})_{\mathcal{L}\mathcal{U}} &= - \left(\mathbf{K}_{\mathcal{L}\mathcal{L}} - \mathbf{K}_{\mathcal{U}\mathcal{L}}^{\top} \mathbf{K}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}} \right)^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^{\top} \mathbf{K}_{\mathcal{U}\mathcal{U}}^{-1} \\
 (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{L}} &= - \left(\mathbf{K}_{\mathcal{U}\mathcal{U}} - \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^{\top} \right)^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \\
 (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} &= \left(\mathbf{K}_{\mathcal{U}\mathcal{U}} - \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^{\top} \right)^{-1}
 \end{aligned} \tag{B.1}$$

Proposition B.1 can be easily proved taken into account that $\mathbf{K}^{-1}\mathbf{K} = \mathbf{I}_n$. This is equivalent to:

$$\begin{bmatrix} (\mathbf{K}^{-1})_{\mathcal{L}\mathcal{L}} & (\mathbf{K}^{-1})_{\mathcal{L}\mathcal{U}} \\ (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{L}} & (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \end{bmatrix} \begin{bmatrix} \mathbf{K}_{\mathcal{L}\mathcal{L}} & \mathbf{K}_{\mathcal{U}\mathcal{L}}^{\top} \\ \mathbf{K}_{\mathcal{U}\mathcal{L}} & \mathbf{K}_{\mathcal{U}\mathcal{U}} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_l & \mathbf{O}_{l \times u} \\ \mathbf{O}_{u \times l} & \mathbf{I}_u \end{bmatrix}$$

From the kernel expansions of Belkin, Niyogi and Sindhwani (2006) for multi-class problems, we have $\mathbf{F} = \mathbf{K}\alpha$ with $\alpha := \begin{bmatrix} \alpha_{\mathcal{L}} \\ \alpha_{\mathcal{U}} \end{bmatrix} \in \mathbb{R}^{n \times c}$. Assuming that $\mathbf{F}_{\mathcal{L}} = \mathbf{Y}_{\mathcal{L}}$, we have:

$$\begin{bmatrix} \mathbf{F}_{\mathcal{L}} \\ \mathbf{F}_{\mathcal{U}} \end{bmatrix} = \begin{bmatrix} \mathbf{K}_{\mathcal{L}\mathcal{L}} & \mathbf{K}_{\mathcal{U}\mathcal{L}}^{\top} \\ \mathbf{K}_{\mathcal{U}\mathcal{L}} & \mathbf{K}_{\mathcal{U}\mathcal{U}} \end{bmatrix} \begin{bmatrix} \alpha_{\mathcal{L}} \\ \alpha_{\mathcal{U}} \end{bmatrix} = \begin{bmatrix} \mathbf{K}_{\mathcal{L}\mathcal{L}}\alpha_{\mathcal{L}} + \mathbf{K}_{\mathcal{U}\mathcal{L}}^{\top}\alpha_{\mathcal{U}} \\ \mathbf{K}_{\mathcal{U}\mathcal{L}}\alpha_{\mathcal{L}} + \mathbf{K}_{\mathcal{U}\mathcal{U}}\alpha_{\mathcal{U}} \end{bmatrix} = \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ \mathbf{F}_{\mathcal{U}} \end{bmatrix} \tag{B.2}$$

Therefore,

$$\alpha_{\mathcal{L}} = \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \left(\mathbf{Y}_{\mathcal{L}} - \mathbf{K}_{\mathcal{U}\mathcal{L}}^{\top} \alpha_{\mathcal{U}} \right) \quad (\text{B.3})$$

From (B.2) and (B.3), we obtain:

$$\mathbf{F}_{\mathcal{U}} = \mathbf{K}_{\mathcal{U}\mathcal{L}} \alpha_{\mathcal{L}} + \mathbf{K}_{\mathcal{U}\mathcal{U}} \alpha_{\mathcal{U}} = \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{Y}_{\mathcal{L}} + (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \alpha_{\mathcal{U}} \quad (\text{B.4})$$

As in (2.5), GFHF can be viewed as the following optimization problem:

$$\min_{\mathbf{F}_{\mathcal{U}} \in \mathbb{R}^{u \times c}} \text{tr} \left(\mathbf{F}_{\mathcal{U}}^{\top} \mathbf{L}_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}} \right) + 2\text{tr} \left(\mathbf{F}_{\mathcal{U}}^{\top} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \right) \quad (\text{B.5})$$

From (B.4), we have:

$$\text{tr} \left(\mathbf{F}_{\mathcal{U}}^{\top} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \right) = \text{tr} \left(\alpha_{\mathcal{U}}^{\top} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \right) + \text{tr} \left(\mathbf{Y}_{\mathcal{L}}^{\top} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^{\top} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \right)$$

and

$$\begin{aligned} \text{tr} \left(\mathbf{F}_{\mathcal{U}}^{\top} \mathbf{L}_{\mathcal{U}\mathcal{U}} \mathbf{F}_{\mathcal{U}} \right) &= \text{tr} \left(\mathbf{Y}_{\mathcal{L}}^{\top} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^{\top} \mathbf{L}_{\mathcal{U}\mathcal{U}} \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{Y}_{\mathcal{L}} \right) \\ &\quad + \text{tr} \left(\alpha_{\mathcal{U}}^{\top} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{U}} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \alpha_{\mathcal{U}} \right) \\ &\quad + 2\text{tr} \left(\alpha_{\mathcal{U}}^{\top} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{U}} \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{Y}_{\mathcal{L}} \right) \end{aligned}$$

Therefore, minimization problem (2.5) becomes:

$$\min_{\alpha_{\mathcal{U}} \in \mathbb{R}^{u \times c}} \Gamma(\alpha_{\mathcal{U}}) \quad (\text{B.6})$$

such that

$$\begin{aligned} \Gamma(\alpha_{\mathcal{U}}) &= \text{tr} \left(\alpha_{\mathcal{U}}^{\top} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{U}} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \alpha_{\mathcal{U}} \right) + 2\text{tr} \left(\alpha_{\mathcal{U}}^{\top} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \right) \\ &\quad + 2\text{tr} \left(\alpha_{\mathcal{U}}^{\top} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{U}} \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{Y}_{\mathcal{L}} \right) \end{aligned}$$

Since (B.6) is a convex optimization problem, we obtain the following closed-form solution:

$$\begin{aligned} \alpha_{\mathcal{U}} &= -(\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} (\mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}}) \mathbf{Y}_{\mathcal{L}} \\ &= (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} - (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \end{aligned} \quad (\text{B.7})$$

Substituting (B.7) in (B.3), we obtain:

$$\begin{aligned}\alpha_{\mathcal{L}} &= \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{Y}_{\mathcal{L}} - \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{L}\mathcal{U}}^{\top} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{L}\mathcal{U}}^{\top} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ &= \left[\mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} + \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{L}\mathcal{U}}^{\top} \mathbf{K}_{\mathcal{U}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}} (\mathbf{K}^{-1})_{\mathcal{L}\mathcal{L}} \right] \mathbf{Y}_{\mathcal{L}} - (\mathbf{K}^{-1})_{\mathcal{L}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}\end{aligned}\quad (\text{B.8})$$

From the first equation in (B.1), we have:

$$\mathbf{K}_{\mathcal{U}\mathcal{L}}^{\top} \mathbf{K}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}} = \mathbf{K}_{\mathcal{L}\mathcal{L}} - (\mathbf{K}^{-1})_{\mathcal{L}\mathcal{L}}^{-1}$$

Therefore, (B.8) yields:

$$\alpha_{\mathcal{L}} = (\mathbf{K}^{-1})_{\mathcal{L}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} - (\mathbf{K}^{-1})_{\mathcal{L}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \quad (\text{B.9})$$

From (B.7) and (B.9), we have:

$$\begin{aligned}\alpha &= \begin{bmatrix} (\mathbf{K}^{-1})_{\mathcal{L}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} - (\mathbf{K}^{-1})_{\mathcal{L}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} - (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \end{bmatrix} \\ &= \begin{bmatrix} (\mathbf{K}^{-1})_{\mathcal{L}\mathcal{L}} & (\mathbf{K}^{-1})_{\mathcal{L}\mathcal{U}} \\ (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{L}} & (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \end{bmatrix} \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \end{bmatrix} \\ &= \mathbf{K}^{-1} \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \end{bmatrix}\end{aligned}\quad (\text{B.10})$$

Eq. (B.10) shows that K-GFHF and GFHF have the same transductive solution, independent of $\mathbf{K}$. However, K-GFHF has the advantage of being an inductive method. Given $\mathbf{x} \in \mathbb{R}^d$, we obtain its output vector as in (2.17).

B.1 Iterative solution for K-GFHF

Computing (B.7) directly might be expensive for data sets with moderate size due to the inversion of a u -by- u matrix. Proposition B.2 provides an iterative solution to compute (B.7) efficiently.

Proposition B.2 (Iterative K-GFHF). *Given an initial matrix $\alpha_{\mathcal{U}}^{(0)} \in \mathbb{R}^{u \times c}$, if the eigenvalues of $\mathbf{I}_u - \mathbf{K}_{\mathcal{U}\mathcal{U}} + \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{L}\mathcal{U}}^{\top}$ are in the interval $[-1, 1]$, (B.6) can be solved by the following recursion:*

$$\alpha_{\mathcal{U}}^{(t+1)} = \left(\mathbf{I}_u - \mathbf{K}_{\mathcal{U}\mathcal{U}} + \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{L}\mathcal{U}}^{\top} \right) \alpha_{\mathcal{U}}^{(t)} + \left(-\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} - \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \right) \mathbf{Y}_{\mathcal{L}} \quad (\text{B.11})$$

Proof. Iterating (B.11), we obtain:

$$\begin{aligned} \alpha_{\mathcal{U}}^{(t)} &= \left(\mathbf{I}_u - \mathbf{K}_{\mathcal{U}\mathcal{U}} + \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^\top \right)^t \alpha_{\mathcal{U}}^{(0)} \\ &+ \left(\sum_{i=0}^{t-1} \left(\mathbf{I}_u - \mathbf{K}_{\mathcal{U}\mathcal{U}} + \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^\top \right)^i \right) \left(-\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} - \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \right) \mathbf{Y}_{\mathcal{L}} \end{aligned} \quad (\text{B.12})$$

If the eigenvalues of $\mathbf{I}_u - \mathbf{K}_{\mathcal{U}\mathcal{U}} + \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^\top$ are in the interval $[-1, 1]$, we have:

$$\lim_{t \rightarrow \infty} \left(\mathbf{I}_u - \mathbf{K}_{\mathcal{U}\mathcal{U}} + \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^\top \right)^t = \mathbf{O}_{u \times u} \quad (\text{B.13})$$

and

$$\lim_{t \rightarrow \infty} \sum_{i=0}^{t-1} \left(\mathbf{I}_u - \mathbf{K}_{\mathcal{U}\mathcal{U}} + \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^\top \right)^i = \left(\mathbf{K}_{\mathcal{U}\mathcal{U}} - \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^\top \right)^{-1} \quad (\text{B.14})$$

Therefore, we have:

$$\lim_{t \rightarrow +\infty} \alpha_{\mathcal{U}}^{(t)} = \left(\mathbf{K}_{\mathcal{U}\mathcal{U}} - \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^\top \right)^{-1} \left(-\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} - \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \right) \mathbf{Y}_{\mathcal{L}}$$

This is equivalent to (B.7) after applying Proposition B.1. $\square$ $\square$

The algorithm in (B.11) usually converges in a few iterations. However, it runs in $\mathcal{O}(u^3)$ time due to the computation of $\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1}$. Lemma B.3 provides iterative solutions to compute $\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1}$ for $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$ and $\mathbf{L} = \mathbf{L}_{\mathbb{N}}$, respectively. For the proof, assume the partitioning

$$\mathbf{D} = \begin{bmatrix} \mathbf{D}_{\mathcal{L}} & \mathbf{O}_{l \times u} \\ \mathbf{O}_{u \times l} & \mathbf{D}_{\mathcal{U}} \end{bmatrix}.$$

Lemma B.3 (Computing $\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1}$ efficiently). *Let $\Theta^{(0)} \in \mathbb{R}^{u \times u}$ be an initial matrix. The iterative method*

$$\Theta^{(t+1)} = (\mathbf{I}_u - \mathbf{D}_{\mathcal{U}} + \mathbf{W}_{\mathcal{U}\mathcal{U}}) \Theta^{(t)} + \mathbf{I}_u$$

converges to $(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1}$, independent of $\Theta^{(0)}$. Moreover, the iterative method

$$\Theta^{(t+1)} = \mathbf{D}_{\mathcal{U}}^{-1/2} \mathbf{W}_{\mathcal{U}\mathcal{U}} \mathbf{D}_{\mathcal{U}}^{-1/2} \Theta^{(t)} + \mathbf{I}_u$$

converges to $(\mathbf{L}_{\mathbb{N}})_{\mathcal{U}\mathcal{U}}^{-1}$, independent of $\Theta^{(0)}$.

Proof. By construction, the eigenvalues of $\mathbf{I}_u - \mathbf{D}_{\mathcal{U}} + \mathbf{W}_{\mathcal{U}\mathcal{U}}$ and $\mathbf{D}_{\mathcal{U}}^{-1/2} \mathbf{W}_{\mathcal{U}\mathcal{U}} \mathbf{D}_{\mathcal{U}}^{-1/2}$ are in the interval $[-1, 1]$ (see (CHUNG, 1997)). The remainder of the proof is equivalent to the proof of Proposition B.2. $\square$ $\square$

We found no theoretical property concerning the interval of the eigenvalues of $\mathbf{I}_u - \mathbf{K}_{\mathcal{U}\mathcal{U}} + \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^\top$. However, the algorithm in (B.11) is effective in practice, with fast convergence.

B.2 Speeding up K-GFHF using Krylov subspaces

Mahdaviani *et al.* (2005) proposed to compute an approximated solution for GFHF for binary classification problems using *Krylov* methods combined with MINRES algorithm (DEMME, 1997; GOLUB; LOAN, 1996). Such an approach can also be applied to K-GFHF.

From (B.6), we can formulate K-GFHF for binary classification problems as follows:

$$\min_{\alpha_{\mathcal{U}} \in \mathbb{R}^u} \Gamma(\alpha_{\mathcal{U}}) \quad (\text{B.15})$$

such that

$$\begin{aligned} \Gamma(\alpha_{\mathcal{U}}) = & \text{tr} \left(\alpha_{\mathcal{U}}^\top (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{U}} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \alpha_{\mathcal{U}} \right) + 2\text{tr} \left(\alpha_{\mathcal{U}}^\top (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{y}_{\mathcal{L}} \right) \\ & + 2\text{tr} \left(\alpha_{\mathcal{U}}^\top (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{U}} \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{y}_{\mathcal{L}} \right) \end{aligned}$$

Therefore, we get the following closed-form solution:

$$\alpha_{\mathcal{U}} = -(\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} (\mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}}) \mathbf{y}_{\mathcal{L}} \quad (\text{B.16})$$

Consider a matrix $\mathbf{Z}$, such that

$$\mathbf{Z} := ((\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}})^{-1} = \mathbf{K}_{\mathcal{U}\mathcal{U}} - \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^\top$$

and a vector $\mathbf{b} = -(\mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}}) \mathbf{y}_{\mathcal{L}}$. Therefore, we have to solve $\mathbf{Z} \alpha_{\mathcal{U}} = \mathbf{b}$. From these definitions, the associated Krylov matrix is given by:

$$\mathbf{K}_r = [\mathbf{b}, \mathbf{Z}\mathbf{b}, \mathbf{Z}^2\mathbf{b}, \dots] \quad (\text{B.17})$$

The *Krylov subspaces* are the spaces spanned by the column vectors of $\mathbf{K}_r$. Since $\mathbf{K}_r$ is a *badly conditioned matrix*¹, we can not find a new estimate $\alpha_{\mathcal{U}}^{(t)}$ by projecting it onto the Krylov subspace. To overcome this issue, we need to construct a well-conditioned *orthogonal matrix*² $\mathbf{Q}^{(t)} := [\mathbf{q}^{(1)}, \dots, \mathbf{q}^{(t)}] \in \mathbb{R}^{u \times t}$ that spans the Krylov space at iteration t . That is, the first t columns of $\mathbf{K}_r$ and $\mathbf{Q}^{(t)}$ span the same space. This can be done, in the *Gram-Schmidt process*³, using the following recurrence (see (DEMME, 1997; GOLUB; LOAN, 1996)):

$$\beta^{(t)} \mathbf{q}^{(t+1)} = \mathbf{Z} \mathbf{q}^{(t)} - \beta^{(t-1)} \mathbf{q}^{(t-1)} - \zeta^{(t)} \mathbf{q}^{(t)} \quad (\text{B.18})$$

where $\beta^{(t)} \in \mathbb{R}$ is the normalizing constant at iteration t and $\zeta^{(t)} \in \mathbb{R}$ is estimated as $\zeta^{(t)} = (\mathbf{q}^{(t)})^\top \mathbf{Z} \mathbf{q}^{(t)}$. From (B.18), we can write $\mathbf{Z} \mathbf{Q}^{(t)} = \mathbf{Q}^{(t+1)} \tilde{\mathbf{H}}^{(t)}$ where $\tilde{\mathbf{H}}^{(t)}$ is the following *tridiagonal matrix*⁴:

$$\tilde{\mathbf{H}}^{(t)} = \begin{pmatrix} \zeta^{(1)} & \beta^{(1)} & 0 & \dots & 0 & 0 \\ \beta^{(1)} & \zeta^{(2)} & \beta^{(2)} & \dots & 0 & 0 \\ 0 & \beta^{(2)} & \zeta^{(3)} & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \zeta^{(t-1)} & \beta^{(t-1)} \\ 0 & 0 & 0 & \dots & \beta^{(t-1)} & \zeta^{(t)} \end{pmatrix}$$

Applying the MINRES algorithm to solve $\mathbf{Z} \alpha_{\mathcal{U}} = \mathbf{b}$ leads to the following optimization problem at iteration t :

$$\min_{\mathbf{c} \in \mathbb{R}^t} \left\| \tilde{\mathbf{H}}^{(t)} \mathbf{c} - \|\mathbf{b}\|_2 \mathbf{1}_n \right\|_2 \quad (\text{B.19})$$

After solving (B.19), we obtain the solution $\alpha_{\mathcal{U}}^{(t)} = \mathbf{Q}^{(t)} \mathbf{c}$ at iteration t . The MINRES algorithm to compute the solution of K-GFHF is summarized in Alg. 5. After approximating the value of $\alpha_{\mathcal{U}}$, we use (B.3) to compute $\alpha_{\mathcal{L}}$.

¹ The condition number of a function with respect to an argument measures how much the output value of the function can change for a small change in the input argument. Specifically, the condition number associated with the linear equation $\mathbf{A} \mathbf{x} = \mathbf{b}$ gives a *bound* on how *inaccurate* the solution $\mathbf{x}$ will be after approximation. Therefore, a matrix with large condition number is badly-conditioned.

² A matrix $\mathbf{A} \in \mathbb{R}^{a \times a}$ is orthogonal if $\mathbf{A} \mathbf{A}^\top = \mathbf{A}^\top \mathbf{A} = \mathbf{I}_a$. Equivalently, we have $\mathbf{A}^\top = \mathbf{A}^{-1}$.

³ Let the projection operator defined by $\text{proj}_{\mathbf{u}}(\mathbf{v}) := \frac{\langle \mathbf{v}, \mathbf{u} \rangle}{\langle \mathbf{u}, \mathbf{u} \rangle} \mathbf{u}$ where $\langle \mathbf{v}, \mathbf{u} \rangle$ denotes the inner product of the vectors $\mathbf{v}$ and $\mathbf{u}$. Given a sequence of vectors $\{\mathbf{v}_1, \dots, \mathbf{v}_k\}$, the Gram-Schmidt process generates the sequence of orthogonal vectors $\{\mathbf{u}_1, \dots, \mathbf{u}_k\}$ such that $\mathbf{u}_k = \mathbf{v}_k - \sum_{j=1}^{k-1} \text{proj}_{\mathbf{u}_j}(\mathbf{v}_k)$.

⁴ A tridiagonal matrix is a square matrix that is both upper and lower Hessenberg. Mathematically, a matrix $\mathbf{A} \in \mathbb{R}^{a \times a}$ is tridiagonal if $\mathbf{A}_{i,j} = 0$ for $|i - j| > 1$.

Algorithm 5 – MINRES algorithm for K-GFHF

Input: $\mathbf{b} \in \mathbb{R}^u$ and $\mathbf{Z} \in \mathbb{R}^{u \times u}$
 $\beta^{(0)} \leftarrow 0$, $\mathbf{q}^{(0)} \leftarrow \mathbf{0}_u$, $\mathbf{q}^{(1)} \leftarrow \mathbf{b}/\|\mathbf{b}\|$
for $t \leftarrow 1$ **to** $t_{\max}$ **do**
 $\mathbf{v} \leftarrow \mathbf{Z}\mathbf{q}^{(t)}$, $\zeta^{(t)} \leftarrow \left(\mathbf{q}^{(t)}\right)^\top \mathbf{v}$
 $\mathbf{v} \leftarrow \mathbf{v} - \beta^{(t-1)}\mathbf{q}^{(t-1)} - \zeta^{(t)}\mathbf{q}^{(t)}$
 $\beta^{(t)} \leftarrow \|\mathbf{v}\|_2$, $\mathbf{q}^{(t+1)} \leftarrow \mathbf{v}/\beta^{(t)}$
 Find $\mathbf{c}$ that solves (B.19) and set $\alpha_{\mathcal{U}}^{(t)} = \mathbf{Q}^{(t)}\mathbf{c}$
end for
Output: $\alpha_{\mathcal{U}}^{(t_{\max})}$

KERNEL APPROACH FOR CGFHF

This chapter proposes a kernel approach for CGFHF. We call such an approach *Kernelized CGFHF* (K-CGFHF). K-CGFHF has the same objective function in (B.6), incorporating the normalization constraints in (LIU; CHANG, 2009). The normalization constraints $\mathbf{F}\mathbf{1}_c = \mathbf{1}_n$ and $\mathbf{F}^\top \mathbf{1}_n = n\omega$, subject to $\mathbf{F}_{\mathcal{L}} = \mathbf{Y}_{\mathcal{L}}$, are equivalent to:

$$\mathbf{F}_{\mathcal{U}}\mathbf{1}_c = \mathbf{1}_u, \quad \mathbf{F}_{\mathcal{U}}^\top \mathbf{1}_u = n\omega - \mathbf{Y}_{\mathcal{L}}^\top \mathbf{1}_l \quad (\text{C.1})$$

From (B.4) and (C.1), we get the following constraints:

$$\begin{aligned} \alpha_{\mathcal{U}}\mathbf{1}_c &= (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} (\mathbf{1}_u - \mathbf{K}_{\mathcal{U}\mathcal{L}}\mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1}\mathbf{1}_l) \\ \alpha_{\mathcal{U}}^\top (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u &= n\omega - \mathbf{Y}_{\mathcal{L}}^\top \mathbf{1}_l - \mathbf{Y}_{\mathcal{L}}^\top \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^\top \mathbf{1}_u \end{aligned} \quad (\text{C.2})$$

From (B.6) and (C.2), we can formulate K-CGFHF as the following constrained optimization problem:

$$\begin{aligned} &\min_{\alpha_{\mathcal{U}} \in \mathbb{R}^{u \times c}} \Gamma(\alpha_{\mathcal{U}}) \\ \text{s.t. } &\begin{cases} \alpha_{\mathcal{U}}\mathbf{1}_c = (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} (\mathbf{1}_u - \mathbf{K}_{\mathcal{U}\mathcal{L}}\mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1}\mathbf{1}_l) \\ \alpha_{\mathcal{U}}^\top (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u = n\omega - \mathbf{Y}_{\mathcal{L}}^\top \mathbf{1}_l - \mathbf{Y}_{\mathcal{L}}^\top \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^\top \mathbf{1}_u \end{cases} \end{aligned} \quad (\text{C.3})$$

Proposition C.1 provides the closed-form solution of (C.3) while Corollary C.2 proves that K-CGFHF generalizes RMGT and RMGTHOR, maintaining the same transductive solution, independently of $\mathbf{K}$.

Proposition C.1 (*Solution of K-CGFHF*). *The closed-form solution of (C.3) is given by:*

$$\alpha_{\mathcal{U}} = (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \left[-\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \tau + \frac{1}{c} \mathbf{v} \mathbf{1}_c^\top \right] \quad (\text{C.4})$$

where

$$\begin{aligned}\tau &= n\omega^\top - \mathbf{1}_l^\top \mathbf{Y}_{\mathcal{L}} + \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ \mathbf{v} &= \mathbf{1}_u + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l - \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \left(u + \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l \right)\end{aligned}$$

Proof. The Lagrangian corresponding to (C.3) is given by:

$$\begin{aligned}\mathcal{L}(\alpha_{\mathcal{U}}, \xi, \lambda) &= \Gamma(\alpha_{\mathcal{U}}) - \xi^\top (\alpha_{\mathcal{U}} \mathbf{1}_c - (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} (\mathbf{1}_u - \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{1}_l)) \\ &\quad - \lambda^\top \left(\alpha_{\mathcal{U}}^\top (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u - n\omega + \mathbf{Y}_{\mathcal{L}}^\top \mathbf{1}_l + \mathbf{Y}_{\mathcal{L}}^\top \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{K}_{\mathcal{U}\mathcal{L}}^\top \mathbf{1}_u \right)\end{aligned}$$

where $\xi \in \mathbb{R}^u$ and $\lambda \in \mathbb{R}^c$ are the Lagrange multipliers. Zeroing $\partial \mathcal{L} / \partial \alpha_{\mathcal{U}}$, we obtain:

$$\alpha_{\mathcal{U}} = (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \left[-(\mathbf{L}_{\mathcal{U}\mathcal{U}} \mathbf{K}_{\mathcal{U}\mathcal{L}} \mathbf{K}_{\mathcal{L}\mathcal{L}}^{-1} + \mathbf{L}_{\mathcal{U}\mathcal{L}}) \mathbf{Y}_{\mathcal{L}} + \frac{1}{2} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \xi \mathbf{1}_c^\top + \frac{1}{2} \mathbf{1}_u \lambda^\top \right] \quad (\text{C.5})$$

Substituting (C.5) in the first constraint, we obtain:

$$\frac{c}{2} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \xi = \mathbf{1}_u + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l - \frac{1}{2} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u \lambda^\top \mathbf{1}_c \quad (\text{C.6})$$

Substituting (C.5) in the second constraint, we obtain:

$$\lambda^\top = \frac{2}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \left[n\omega^\top - \mathbf{1}_l^\top \mathbf{Y}_{\mathcal{L}} + \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} - \frac{1}{2} \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \xi \mathbf{1}_c \right] \quad (\text{C.7})$$

Substituting (C.7) in (C.6), we obtain:

$$\xi = \frac{2}{c} \varpi^{-1} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \left[\mathbf{1}_u + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l - \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \left(u + \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l \right) \right] \quad (\text{C.8})$$

in which

$$\varpi = (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} - \frac{(\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}}}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}$$

Substituting (C.7) in (C.5), we obtain:

$$\begin{aligned}\alpha_{\mathcal{U}} &= -(\mathbf{K}^{-1})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} - (\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ &\quad + \frac{1}{2} \varpi \xi \mathbf{1}_c^\top + \frac{(\mathbf{K}^{-1})_{\mathcal{U}\mathcal{U}} \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \left(n\omega^\top - \mathbf{1}_l^\top \mathbf{Y}_{\mathcal{L}} + \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \right)\end{aligned} \quad (\text{C.9})$$

Substituting (C.8) in (C.9), we get (C.4). $\square$

Corollary C.2. *From (C.4) and (B.3), we have:*

$$\alpha = \mathbf{K}^{-1} \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \tau + \frac{1}{c} \mathbf{v} \mathbf{1}_c^\top \end{bmatrix} \quad (\text{C.10})$$

in which

$$\begin{aligned} \tau &= n\omega^\top - \mathbf{1}_l^\top \mathbf{Y}_{\mathcal{L}} + \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ \mathbf{v} &= \mathbf{1}_u + \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l - \frac{\mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \left(u + \mathbf{1}_u^\top \mathbf{L}_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{1}_l \right) \end{aligned}$$

Moreover, if $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$, (C.10) yields:

$$\alpha = \mathbf{K}^{-1} \begin{bmatrix} \mathbf{Y}_{\mathcal{L}} \\ -(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} + \frac{(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u}{\mathbf{1}_u^\top (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} \mathbf{1}_u} \tau \end{bmatrix} \quad (\text{C.11})$$

in which

$$\tau = n\omega^\top - \mathbf{1}_l^\top \mathbf{Y}_{\mathcal{L}} + \mathbf{1}_u^\top (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}}^{-1} (\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}}$$

Proof. Eq. (C.10) comes directly from (C.4) and (B.3). From (C.10) and Corollary 4.2, we get (C.11). $\square$

HEURISTIC INDUCTIVE SOLUTION FOR LGC

This chapter proposes a new matrix formulation of LGC under certain constraints on the training sample. We call this formulation *Reformulated Local and Global Consistency* (RLGC). We start our analysis rewriting the general solution of LGC. Then, we provide approximations of this solution from a data sparseness assumption.

D.1 Reformulation of LGC

From the transductive solution in (2.12), we can write:

$$\begin{bmatrix} \mathbf{L}_{\mathcal{L}\mathcal{L}} + \Sigma_{\mathcal{L}} & \mathbf{L}_{\mathcal{U}\mathcal{L}}^\top \\ \mathbf{L}_{\mathcal{U}\mathcal{L}} & \mathbf{L}_{\mathcal{U}\mathcal{U}} + \Sigma_{\mathcal{U}} \end{bmatrix} \begin{bmatrix} \mathbf{F}_{\mathcal{L}} \\ \mathbf{F}_{\mathcal{U}} \end{bmatrix} = \begin{bmatrix} \Sigma_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ \mathbf{O}_{u \times c} \end{bmatrix}$$

This is equivalent to:

$$\begin{cases} (\mathbf{L}_{\mathcal{L}\mathcal{L}} + \Sigma_{\mathcal{L}}) \mathbf{F}_{\mathcal{L}} + \mathbf{L}_{\mathcal{U}\mathcal{L}}^\top \mathbf{F}_{\mathcal{U}} = \Sigma_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \\ \mathbf{L}_{\mathcal{U}\mathcal{L}} \mathbf{F}_{\mathcal{L}} + (\mathbf{L}_{\mathcal{U}\mathcal{U}} + \Sigma_{\mathcal{U}}) \mathbf{F}_{\mathcal{U}} = \mathbf{O}_{u \times c} \end{cases}$$

Therefore, we have:

$$\mathbf{F}_{\mathcal{L}} = (\mathbf{L}_{\mathcal{L}\mathcal{L}} + \Sigma_{\mathcal{L}})^{-1} (\Sigma_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}} - \mathbf{L}_{\mathcal{U}\mathcal{L}}^\top \mathbf{F}_{\mathcal{U}}) \quad (\text{D.1})$$

$$\mathbf{F}_{\mathcal{U}} = -\Gamma \mathbf{L}_{\mathcal{U}\mathcal{L}} (\mathbf{L}_{\mathcal{L}\mathcal{L}} + \Sigma_{\mathcal{L}})^{-1} \Sigma_{\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \quad (\text{D.2})$$

in which

$$\Gamma = \left(\mathbf{L}_{\mathcal{U}\mathcal{U}} + \Sigma_{\mathcal{U}} - \mathbf{L}_{\mathcal{U}\mathcal{L}} (\mathbf{L}_{\mathcal{L}\mathcal{L}} + \Sigma_{\mathcal{L}})^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}}^{\top} \right)^{-1} \quad (\text{D.3})$$

Eq. (D.1) and (D.2) are the general solutions of LGC for the labeled and unlabeled examples, respectively. Corollary D.1 shows a simple iterative method to compute Γ efficiently for a given graph Laplacian $\mathbf{L}$.

Corollary D.1 (Computing Γ iteratively). *Given an initial matrix $\mathbf{X}^{(0)} \in \mathbb{R}^{u \times u}$, if the eigenvalues of $\mathbf{I}_u - \Gamma^{-1}$ are in the interval $[-1, 1]$, we can compute (D.3) using the following recursion:*

$$\mathbf{X}^{(t+1)} = \left(\mathbf{I}_u - \mathbf{L}_{\mathcal{U}\mathcal{U}} - \Sigma_{\mathcal{U}} + \mathbf{L}_{\mathcal{U}\mathcal{L}} (\mathbf{L}_{\mathcal{L}\mathcal{L}} + \Sigma_{\mathcal{L}})^{-1} \mathbf{L}_{\mathcal{U}\mathcal{L}}^{\top} \right) \mathbf{X}^{(t)} + \mathbf{I}_u \quad (\text{D.4})$$

independently of $\mathbf{X}^{(0)}$. More precisely, $\lim_{t \rightarrow \infty} \mathbf{X}^{(t)} = \Gamma$.

Proof. The proof is similar to the proof of Proposition 3.4. □

Let us define the following n -by- n matrix:

$$\Upsilon := \mathbf{I}_n - \frac{1}{1+\mu} \mathbf{S} = \mathbf{I}_n - \frac{1}{1+\mu} \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2}$$

Proposition D.2 provides useful properties of Υ for our task. Specifically, it proves that Υ is PSD and *diagonally dominant*¹ under some constraints. The diagonally dominance of Υ means that we can solve linear systems in Υ in *nearly-linear time*² (SPIELMAN; TENG, 2006).

Proposition D.2 (Properties of Υ). *Υ has the following properties:*

1. If, $\forall i \in \mathbb{N}_n$,

$$\sum_{j \neq i} \sqrt{\mathbf{D}_{jj}} \leq (1+\mu) \sqrt{\mathbf{D}_{ii}} \quad (\text{D.5})$$

then Υ is diagonally dominant.

2. If $\mathbf{W} = \mathbf{W}^{\top}$, then Υ is PSD $\forall \mu > 0$.

¹ A matrix $\mathbf{A} \in \mathbb{R}^{a \times a}$ is diagonally dominant if $|\mathbf{A}_{ii}| \geq \sum_{j \neq i} |\mathbf{A}_{ij}|$, $\forall i \in \mathbb{N}_a$.

² Unfortunately, we found no evidence in the literature concerning the empirical effectiveness of the numerical method of Spielman and Teng (2006).

3. Since $\Upsilon = \Upsilon^\top$ and Υ is invertible, we have:

$$\begin{aligned}
 (\Upsilon^{-1})_{\mathcal{L}\mathcal{L}} &= \left(\Upsilon_{\mathcal{L}\mathcal{L}} - \Upsilon_{\mathcal{U}\mathcal{L}}^\top \Upsilon_{\mathcal{U}\mathcal{U}}^{-1} \Upsilon_{\mathcal{U}\mathcal{L}} \right)^{-1} \\
 (\Upsilon^{-1})_{\mathcal{L}\mathcal{U}} &= -(\Upsilon^{-1})_{\mathcal{L}\mathcal{L}} \Upsilon_{\mathcal{U}\mathcal{L}}^\top \Upsilon_{\mathcal{U}\mathcal{U}}^{-1} \\
 (\Upsilon^{-1})_{\mathcal{U}\mathcal{L}} &= -(\Upsilon^{-1})_{\mathcal{U}\mathcal{U}} \Upsilon_{\mathcal{U}\mathcal{L}} \Upsilon_{\mathcal{L}\mathcal{L}}^{-1} \\
 (\Upsilon^{-1})_{\mathcal{U}\mathcal{U}} &= \left(\Upsilon_{\mathcal{U}\mathcal{U}} - \Upsilon_{\mathcal{U}\mathcal{L}} \Upsilon_{\mathcal{L}\mathcal{L}}^{-1} \Upsilon_{\mathcal{U}\mathcal{L}}^\top \right)^{-1}
 \end{aligned} \tag{D.6}$$

Proof. 1. $\forall i \in \mathbb{N}_n$, we have:

$$\begin{aligned}
 \sum_{j \neq i} |\Upsilon_{ij}| &= \sum_{j \neq i} \left| \frac{1}{1+\mu} \frac{\mathbf{W}_{ij}}{\sqrt{\mathbf{D}_{ii}} \sqrt{\mathbf{D}_{jj}}} \right| = \frac{1}{1+\mu} \frac{1}{\sqrt{\mathbf{D}_{ii}}} \sum_{j \neq i} \frac{\mathbf{W}_{ij}}{\sqrt{\mathbf{D}_{jj}}} \\
 &= \frac{1}{1+\mu} \frac{1}{\sqrt{\mathbf{D}_{ii}}} \sum_{j \neq i} \frac{\mathbf{W}_{ij}}{\mathbf{D}_{jj}} \sqrt{\mathbf{D}_{jj}} \leq \frac{1}{1+\mu} \frac{1}{\sqrt{\mathbf{D}_{ii}}} \sum_{j \neq i} \sqrt{\mathbf{D}_{jj}}
 \end{aligned}$$

If (D.5) holds, then

$$\sum_{j \neq i} |\Upsilon_{ij}| \leq 1 = |\Upsilon_{ii}|$$

2. $\forall \mathbf{v} \in \mathbb{R}^n$, we have:

$$\mathbf{v}^\top \Upsilon \mathbf{v} = \frac{\mu}{1+\mu} \mathbf{v}^\top \mathbf{v} + \frac{1}{1+\mu} \mathbf{v}^\top \mathbf{L}_{\mathbb{N}} \mathbf{v}$$

If $\mathbf{W} = \mathbf{W}^\top$, then $\mathbf{v}^\top \mathbf{L}_{\mathbb{N}} \mathbf{v} \geq 0$. Therefore, if $\mu > 0$, then $\mathbf{v}^\top \Upsilon \mathbf{v} \geq 0$.

3. (D.6) is a well known block matrix inversion property for symmetric, invertible matrices. $\square$

Corollary D.3 rewrites the combinatorial and normalized Laplacians with respect to Υ . Such a result is used in Proposition D.4, which rewrites the solution of LGC with respect to Υ for the combinatorial and normalized Laplacians. In Proposition D.4, we consider that $\Sigma = \mu \mathbf{I}_n$, which is assumed in the original formulation of LGC (see (ZHOU *et al.*, 2004)).

Corollary D.3 (*Graph Laplacians with respect to Υ*). *From the definitions of Υ , $\mathbf{L}_{\mathbb{C}}$, and $\mathbf{L}_{\mathbb{N}}$, we can write:*

$$\begin{aligned}
 \mathbf{L}_{\mathbb{N}} &= (1+\mu)\Upsilon - \mu \mathbf{I}_n \\
 \mathbf{L}_{\mathbb{C}} &= (1+\mu)\mathbf{D}^{1/2} \Upsilon \mathbf{D}^{1/2} - \mu \mathbf{D}
 \end{aligned}$$

Therefore,

$$\begin{aligned}
(\mathbf{L}_{\mathbb{N}})_{\mathcal{L}\mathcal{L}} &= (1 + \mu)\Upsilon_{\mathcal{L}\mathcal{L}} - \mu\mathbf{I}_l \\
(\mathbf{L}_{\mathbb{N}})_{\mathcal{U}\mathcal{L}} &= (1 + \mu)\Upsilon_{\mathcal{U}\mathcal{L}} \\
(\mathbf{L}_{\mathbb{N}})_{\mathcal{U}\mathcal{U}} &= (1 + \mu)\Upsilon_{\mathcal{U}\mathcal{U}} - \mu\mathbf{I}_u
\end{aligned}$$

and

$$\begin{aligned}
(\mathbf{L}_{\mathbb{C}})_{\mathcal{L}\mathcal{L}} &= (1 + \mu)\mathbf{D}_{\mathcal{L}}^{-1/2}\Upsilon_{\mathcal{L}\mathcal{L}}\mathbf{D}_{\mathcal{L}}^{-1/2} - \mu\mathbf{D}_{\mathcal{L}} \\
(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{L}} &= (1 + \mu)\mathbf{D}_{\mathcal{U}}^{-1/2}\Upsilon_{\mathcal{U}\mathcal{L}}\mathbf{D}_{\mathcal{L}}^{-1/2} \\
(\mathbf{L}_{\mathbb{C}})_{\mathcal{U}\mathcal{U}} &= (1 + \mu)\mathbf{D}_{\mathcal{U}}^{-1/2}\Upsilon_{\mathcal{U}\mathcal{U}}\mathbf{D}_{\mathcal{U}}^{-1/2} - \mu\mathbf{D}_{\mathcal{U}}
\end{aligned}$$

Proof. The proof comes directly from the definitions of Υ , $\mathbf{L}_{\mathbb{C}}$, and $\mathbf{L}_{\mathbb{N}}$. $\square$

Proposition D.4 (Solution of LGC with respect to Υ). Assume that $\Sigma_{\mathcal{L}} = \mu\mathbf{I}_l$ and $\Sigma_{\mathcal{U}} = \mu\mathbf{I}_u$. If $\mathbf{L} = \mathbf{L}_{\mathbb{C}}$, (D.1), (D.2), and (D.3) can be rewritten as follows:

$$\mathbf{F}_{\mathcal{L}} = \Gamma^{(1)} \left(\mu\mathbf{Y}_{\mathcal{L}} - (1 + \mu)\mathbf{D}_{\mathcal{L}}^{-1/2}\Upsilon_{\mathcal{U}\mathcal{L}}^{\top}\mathbf{D}_{\mathcal{U}}^{-1/2}\mathbf{F}_{\mathcal{U}} \right) \quad (\text{D.7})$$

$$\mathbf{F}_{\mathcal{U}} = -\mu(1 + \mu)\Gamma^{(2)}\mathbf{D}_{\mathcal{U}}^{-1/2}\Upsilon_{\mathcal{U}\mathcal{L}}\mathbf{D}_{\mathcal{L}}^{-1/2}\Gamma^{(1)}\mathbf{Y}_{\mathcal{L}} \quad (\text{D.8})$$

in which

$$\Gamma^{(1)} = \left((1 + \mu)\mathbf{D}_{\mathcal{L}}^{-1/2}\Upsilon_{\mathcal{L}\mathcal{L}}^{-1}\mathbf{D}_{\mathcal{L}}^{-1/2} - \mu\mathbf{D}_{\mathcal{L}} + \mu\mathbf{I}_l \right)^{-1}$$

and

$$\begin{aligned}
\Gamma^{(2)} &= \left((1 + \mu)\mathbf{D}_{\mathcal{U}}^{-1/2}\Upsilon_{\mathcal{U}\mathcal{U}}\mathbf{D}_{\mathcal{U}}^{-1/2} - \mu\mathbf{D}_{\mathcal{U}} + \mu\mathbf{I}_u \right. \\
&\quad \left. - (1 + \mu)^2\mathbf{D}_{\mathcal{U}}^{-1/2}\Upsilon_{\mathcal{U}\mathcal{L}}\mathbf{D}_{\mathcal{L}}^{-1/2}\Gamma^{(1)}\mathbf{D}_{\mathcal{L}}^{-1/2}\Upsilon_{\mathcal{U}\mathcal{L}}^{\top}\mathbf{D}_{\mathcal{U}}^{-1/2} \right)^{-1}
\end{aligned}$$

If $\mathbf{L} = \mathbf{L}_{\mathbb{N}}$, (D.1), (D.2), and (D.3) can be rewritten as follows:

$$\mathbf{F}_{\mathcal{L}} = \Upsilon_{\mathcal{L}\mathcal{L}}^{-1} \left(\frac{\mu}{1 + \mu}\mathbf{Y}_{\mathcal{L}} - \Upsilon_{\mathcal{U}\mathcal{L}}^{\top}\mathbf{F}_{\mathcal{U}} \right) \quad (\text{D.9})$$

$$\mathbf{F}_{\mathcal{U}} = -\frac{\mu}{1 + \mu}\Gamma^{(3)}\Upsilon_{\mathcal{U}\mathcal{L}}\Upsilon_{\mathcal{L}\mathcal{L}}^{-1}\mathbf{Y}_{\mathcal{L}} \quad (\text{D.10})$$

in which

$$\Gamma^{(3)} = \left(\Upsilon_{\mathcal{U}\mathcal{U}} - \Upsilon_{\mathcal{U}\mathcal{L}}\Upsilon_{\mathcal{L}\mathcal{L}}^{-1}\Upsilon_{\mathcal{U}\mathcal{L}}^{\top} \right)^{-1}$$

Proof. The proof comes directly from Corollary D.3 and (D.1), (D.2), and (D.3). $\square$

Since sparse graphs have much less spurious edges between dissimilar examples and tend to give better empirical results than dense graphs on general SSL tasks (ZHU, 2005), we formulate Assumption D.5. Such an assumption is satisfied in many practical applications when we are dealing with a considerable amount of unlabeled examples on sparse graphs. Intuitively, we assume that there is no connection between labeled examples.

Assumption D.5 (*Sparseness of labeled examples*). *The labeled examples are sparsely sampled, being distant from one to another. Then, we consider that the neighborhood of each labeled example contains only unlabeled examples.*

From Assumption D.5, we have $\mathbf{W}_{\mathcal{L}\mathcal{L}} = \mathbf{O}_{l \times l}$. This yields $\Upsilon_{\mathcal{L}\mathcal{L}} = \Upsilon_{\mathcal{L}\mathcal{L}}^{-1} = \mathbf{I}_l$. Since the normalized Laplacian $\mathbf{L}_{\mathbb{N}}$ can achieve better results than the combinatorial Laplacian $\mathbf{L}_{\mathbb{C}}$ (JOHNSON; ZHANG, 2007), we continue our analysis from (D.9) and (D.10). Since $\Upsilon_{\mathcal{L}\mathcal{L}}^{-1} = \mathbf{I}_l$, (D.9) and (D.10) yield:

$$\mathbf{F}_{\mathcal{L}} = \frac{\mu}{1+\mu} \mathbf{Y}_{\mathcal{L}} - \Upsilon_{\mathcal{U}\mathcal{L}}^{\top} \mathbf{F}_{\mathcal{U}} \quad (\text{D.11})$$

$$\mathbf{F}_{\mathcal{U}} = -\frac{\mu}{1+\mu} \left(\Upsilon_{\mathcal{U}\mathcal{U}} - \Upsilon_{\mathcal{U}\mathcal{L}} \Upsilon_{\mathcal{L}\mathcal{L}}^{\top} \right)^{-1} \Upsilon_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \quad (\text{D.12})$$

Let $\mathbf{M} := \mathbf{D}^{-1/2} \mathbf{1}_n \in \mathbb{R}^n$. Since $\mathbf{W} = \mathbf{W}^{\top}$, $\mathbf{M}$ can be written as follows:

$$\mathbf{M}_i = \begin{cases} \left(\sum_{j=1}^u (\mathbf{W}_{\mathcal{U}\mathcal{L}})_{ji} \right)^{-1/2}, & i \leq l \\ \left(\sum_{j=1}^l (\mathbf{W}_{\mathcal{U}\mathcal{L}})_{i-l,j} + \sum_{j=1}^u (\mathbf{W}_{\mathcal{U}\mathcal{U}})_{i-l,j} \right)^{-1/2}, & i > l \end{cases}$$

Therefore, we can write:

$$(\Upsilon_{\mathcal{U}\mathcal{U}})_{i-l,j-l} = \begin{cases} -\frac{1}{1+\mu} (\mathbf{W}_{\mathcal{U}\mathcal{U}})_{i-l,j-l} \mathbf{M}_i \mathbf{M}_j, & \mathbf{x}_j \in \mathcal{N}_i \\ 1, & i = j \\ 0, & \text{otherwise} \end{cases}$$

Moreover, we have:

$$(\Upsilon_{\mathcal{U}\mathcal{L}})_{i-l,j} = \begin{cases} -\frac{1}{1+\mu} (\mathbf{W}_{\mathcal{U}\mathcal{L}})_{i-l,j} \mathbf{M}_i \mathbf{M}_j, & \text{if } \mathbf{x}_j \in \mathcal{N}_i \\ 0, & \text{otherwise} \end{cases}$$

Let $\Phi := \Upsilon_{\mathcal{U}\mathcal{U}} - \Upsilon_{\mathcal{U}\mathcal{L}}\Upsilon_{\mathcal{U}\mathcal{L}}^\top \in \mathbb{R}^{u \times u}$. Assuming that the symbol $\langle \cdot, \cdot \rangle_l$ represents the inner product of vectors in $\mathbb{R}^l$, we can write:

$$\Phi_{ij} = \begin{cases} 1 - \sum_{k=1}^l (\Upsilon_{\mathcal{U}\mathcal{L}})_{ik}^2, & i = j \\ (\Upsilon_{\mathcal{U}\mathcal{U}})_{ij} - \langle (\Upsilon_{\mathcal{U}\mathcal{L}})_{i,\cdot}, (\Upsilon_{\mathcal{U}\mathcal{L}})_{j,\cdot} \rangle_l, & i \neq j \end{cases}$$

Then, we rewrite (D.12) as follows:

$$\mathbf{F}_{\mathcal{U}} = -\frac{\mu}{1+\mu} \Phi^{-1} \Upsilon_{\mathcal{U}\mathcal{L}} \mathbf{Y}_{\mathcal{L}} \quad (\text{D.13})$$

Proposition D.6 proves that Φ is *diagonally dominant* when Υ is also diagonally dominant. Mathematically, we have:

$$\forall i \in \mathbb{N}_u, \sum_{j \neq i}^u |\Upsilon_{ij}| \leq |\Upsilon_{ii}| \quad \implies \quad \forall i \in \mathbb{N}_u, \sum_{j \neq i}^u |\Phi_{ij}| \leq |\Phi_{ii}|$$

In this scenario, we may use the numerical method of Spielman and Teng (2006) to speed up the computation of (D.13).

Proposition D.6 (*Diagonal dominance of Φ*). *If Υ is diagonally dominant, then Φ is also diagonally dominant.*

Proof. Assume $i \in \mathbb{N}_u$. From the hypothesis, we have:

$$\sum_{j \neq i}^n |\Upsilon_{i+l,j}| = \sum_{j \neq i}^u |(\Upsilon_{\mathcal{U}\mathcal{U}})_{ij}| + \sum_{j=1}^l |(\Upsilon_{\mathcal{U}\mathcal{L}})_{ij}| \leq |\Upsilon_{i+l,i+l}| = 1$$

This yields:

$$\sum_{j \neq i}^u |(\Upsilon_{\mathcal{U}\mathcal{U}})_{ij}| \leq 1 + \sum_{j=1}^l (\Upsilon_{\mathcal{U}\mathcal{L}})_{ij}$$

We can write:

$$\begin{aligned} \sum_{j \neq i}^u \langle (\Upsilon_{\mathcal{U}\mathcal{L}})_{i,\cdot}, (\Upsilon_{\mathcal{U}\mathcal{L}})_{j,\cdot} \rangle_l &= - \sum_{k=1}^l (\Upsilon_{\mathcal{U}\mathcal{L}})_{ik} \sum_{j \neq i}^u |(\Upsilon_{\mathcal{U}\mathcal{L}})_{jk}| \\ &\leq - \sum_{k=1}^l (\Upsilon_{\mathcal{U}\mathcal{L}})_{ik} (1 - |(\Upsilon_{\mathcal{U}\mathcal{L}})_{ik}|) \end{aligned}$$

Then,

$$\begin{aligned}
\sum_{j \neq i}^u |\Phi_{ij}| &= \sum_{j \neq i}^u |(\Upsilon_{\mathcal{U}\mathcal{L}})_{ij} - \langle (\Upsilon_{\mathcal{U}\mathcal{L}})_{i\cdot}, (\Upsilon_{\mathcal{U}\mathcal{L}})_{j\cdot} \rangle_l| \\
&= \sum_{j \neq i}^u \langle (\Upsilon_{\mathcal{U}\mathcal{L}})_{i\cdot}, (\Upsilon_{\mathcal{U}\mathcal{L}})_{j\cdot} \rangle_l + \sum_{j \neq i}^u |(\Upsilon_{\mathcal{U}\mathcal{L}})_{ij}| \\
&\leq 1 - \sum_{k=1}^l (\Upsilon_{\mathcal{U}\mathcal{L}})_{ik} (1 - |(\Upsilon_{\mathcal{U}\mathcal{L}})_{ik}|) + \sum_{j=1}^l (\Upsilon_{\mathcal{U}\mathcal{L}})_{ij} \\
&\leq 1 - \sum_{k=1}^l (\Upsilon_{\mathcal{U}\mathcal{L}})_{ik}^2 = |\Phi_{ii}|
\end{aligned}$$

□

D.2 iLGC

This section provides an inductive approach for LGC using the reformulation proposed in Section D. Consider the same notation in Section A for the induction process.

Let $\tilde{\mathbf{F}} \in \mathbb{R}^{(n+1) \times c}$ be the output matrix of LGC for the *extended* training sample $\tilde{\mathcal{X}}$ and $\tilde{\mathbf{Y}}_{\mathcal{L}} := \mathbf{F} \in \mathbb{R}^{n \times c}$ the corresponding label matrix. The optimization problem of LGC for $\tilde{\mathcal{X}}$ is given by:

$$\min_{\tilde{\mathbf{F}} \in \mathbb{R}^{(n+1) \times c}} \text{tr} \left(\tilde{\mathbf{F}}^\top \tilde{\mathbf{L}} \tilde{\mathbf{F}} + \left(\tilde{\mathbf{F}} - \tilde{\mathbf{Y}} \right)^\top \tilde{\Sigma} \left(\tilde{\mathbf{F}} - \tilde{\mathbf{Y}} \right) \right)$$

As in (2.12), we obtain the following closed-form solution:

$$\tilde{\mathbf{F}} = \left(\tilde{\mathbf{L}} + \tilde{\Sigma} \right)^{-1} \tilde{\Sigma} \tilde{\mathbf{Y}} \quad (\text{D.14})$$

Using the partitions described in Section D, (D.14) can be rewritten as the following output matrices for labeled and unlabeled examples:

$$\begin{aligned}
\tilde{\mathbf{F}}_{\mathcal{L}} &= \left(\tilde{\mathbf{L}}_{\mathcal{L}\mathcal{L}} + \tilde{\Sigma}_{\mathcal{L}} \right)^{-1} \left(\tilde{\Sigma}_{\mathcal{L}} \mathbf{F} - \tilde{\mathbf{L}}_{\mathcal{U}\mathcal{L}}^\top \tilde{\mathbf{F}}_{\mathcal{U}} \right) \\
\tilde{\mathbf{F}}_{\mathcal{U}} &= -\tilde{\Gamma} \tilde{\mathbf{L}}_{\mathcal{U}\mathcal{L}} \left(\tilde{\mathbf{L}}_{\mathcal{L}\mathcal{L}} + \tilde{\Sigma}_{\mathcal{L}} \right)^{-1} \tilde{\Sigma}_{\mathcal{L}} \mathbf{F}
\end{aligned} \quad (\text{D.15})$$

in which

$$\tilde{\Gamma} = \left(\tilde{\mathbf{L}}_{\mathcal{U}\mathcal{U}} + \tilde{\Sigma}_{\mathcal{U}} - \tilde{\mathbf{L}}_{\mathcal{U}\mathcal{L}} \left(\tilde{\mathbf{L}}_{\mathcal{L}\mathcal{L}} + \tilde{\Sigma}_{\mathcal{L}} \right)^{-1} \tilde{\mathbf{L}}_{\mathcal{U}\mathcal{L}}^\top \right)^{-1}$$

As in Section D, we formulate our inductive approach for LGC using the normalized Laplacian $\mathbf{L}_{\mathbb{N}}$ due to the statistical analysis of Johnson and Zhang (2007). As in Corollary D.3, we can write:

$$\tilde{\mathbf{L}}_{\mathbb{N}} = (1 + \mu)\tilde{\mathbf{Y}} - \mu\mathbf{I}_{n+1}$$

Assuming that $\tilde{\Sigma}_{\mathcal{L}} = \mu\mathbf{I}_{n+1}$, $\tilde{\Sigma}_{\mathcal{U}} = \mu$, and $\tilde{\mathbf{L}} = \tilde{\mathbf{L}}_{\mathbb{N}}$, (D.1) yields:

$$\tilde{\mathbf{F}}_{\mathcal{L}} = \frac{\mu}{1 + \mu}\mathbf{F} - \tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{L}}^{\top}\tilde{\mathbf{F}}_{\mathcal{U}} \quad (\text{D.16})$$

Moreover, (D.2) yields:

$$\tilde{\mathbf{F}}_{\mathcal{U}} = -\frac{\mu}{1 + \mu} \left(\tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{U}} - \tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{L}}\tilde{\mathbf{Y}}_{\mathcal{L}\mathcal{L}}^{-1}\tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{L}}^{\top} \right)^{-1} \tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{L}}\tilde{\mathbf{Y}}_{\mathcal{L}\mathcal{L}}^{-1}\mathbf{F} \quad (\text{D.17})$$

Note that $\tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{U}} - \tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{L}}\tilde{\mathbf{Y}}_{\mathcal{L}\mathcal{L}}^{-1}\tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{L}}^{\top}$ is a real number. Unfortunately, computing (D.17) may be unfeasible even for data sets with moderate sizes due to the inversion of the n -by- n matrix $\tilde{\mathbf{Y}}_{\mathcal{L}\mathcal{L}}$ for each unseen example $\tilde{\mathbf{x}}$. We provide in the following a *linear time approximation* of (D.17) to overcome this issue. Such an approximation is shown in Theorem D.7.

From $\tilde{\mathbf{F}}_{\mathcal{U}}$, we classify $\tilde{\mathbf{x}}$ as:

$$\begin{aligned} y(\tilde{\mathbf{x}}) &= \arg \max_{i \in \mathbb{N}_c} [\tilde{\mathbf{F}}_{\mathcal{U}}]_i \\ &= \arg \max_{i \in \mathbb{N}_c} \left[-\frac{\mu}{1 + \mu} \left(\tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{U}} - \tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{L}}\tilde{\mathbf{Y}}_{\mathcal{L}\mathcal{L}}^{-1}\tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{L}}^{\top} \right)^{-1} \tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{L}}\tilde{\mathbf{Y}}_{\mathcal{L}\mathcal{L}}^{-1}\mathbf{F} \right]_i \\ &= \arg \max_{i \in \mathbb{N}_c} \left[-\left(\tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{U}} - \tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{L}}\tilde{\mathbf{Y}}_{\mathcal{L}\mathcal{L}}^{-1}\tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{L}}^{\top} \right)^{-1} \tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{L}}\tilde{\mathbf{Y}}_{\mathcal{L}\mathcal{L}}^{-1}\mathbf{F} \right]_i \end{aligned} \quad (\text{D.18})$$

From Assumption A.1, $\tilde{\mathbf{W}}_{\mathcal{L}\mathcal{L}} = \mathbf{W}$. Given a similarity function $\mathcal{H}(\cdot, \cdot)$, we compute $\tilde{\mathbf{W}}_{\mathcal{U}\mathcal{L}}^{\top} \in \mathbb{R}^n$ as $(\tilde{\mathbf{W}}_{\mathcal{U}\mathcal{L}}^{\top})_i = \mathcal{H}(\tilde{\mathbf{x}}, \mathbf{x}_i)$, $\forall \mathbf{x}_i \in \tilde{\mathcal{N}}$, and $(\tilde{\mathbf{W}}_{\mathcal{U}\mathcal{L}}^{\top})_i = 0$ otherwise. Let $\tilde{\mathbf{M}} := \tilde{\mathbf{D}}^{-1/2}\mathbf{1}_{n+1} \in \mathbb{R}^{n+1}$. Therefore, we can write:

$$\tilde{\mathbf{M}}_i = \begin{cases} \left(\frac{1}{\mathbf{M}_i^2} + \mathcal{H}(\tilde{\mathbf{x}}, \mathbf{x}_i)\tilde{\mathbf{N}}_i \right)^{-1/2}, & 1 \leq i \leq n \\ \left(\sum_{\mathbf{x}_j \in \tilde{\mathcal{N}}} \mathcal{H}(\tilde{\mathbf{x}}, \mathbf{x}_j) \right)^{-1/2}, & i = n + 1 \end{cases}$$

Then, we compute $\tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{L}}^{\top} \in \mathbb{R}^n$ as follows:

$$\begin{aligned} (\tilde{\mathbf{Y}}_{\mathcal{U}\mathcal{L}}^{\top})_i &= -\frac{1}{1 + \mu} (\tilde{\mathbf{W}}_{\mathcal{U}\mathcal{L}}^{\top})_i \tilde{\mathbf{M}}_i \tilde{\mathbf{M}}_{n+1} \\ &= -\frac{1}{1 + \mu} \frac{\mathcal{H}(\tilde{\mathbf{x}}, \mathbf{x}_i)\tilde{\mathbf{N}}_i \left(\sum_{\mathbf{x}_j \in \tilde{\mathcal{N}}} \mathcal{H}(\tilde{\mathbf{x}}, \mathbf{x}_j) \right)^{-1/2}}{\left(\frac{1}{\mathbf{M}_i^2} + \mathcal{H}(\tilde{\mathbf{x}}, \mathbf{x}_i)\tilde{\mathbf{N}}_i \right)^{-1/2}} \end{aligned}$$

if $\mathbf{x}_i \in \tilde{\mathcal{N}}$ and 0 otherwise. Let $\tilde{\Phi} \in \mathbb{R}$ such that

$$\tilde{\Phi} := \tilde{\Upsilon}_{\mathcal{U}\mathcal{U}} - \tilde{\Upsilon}_{\mathcal{U}\mathcal{L}} \tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1} \tilde{\Upsilon}_{\mathcal{U}\mathcal{L}}^\top = 1 - \tilde{\Upsilon}_{\mathcal{U}\mathcal{L}} \tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1} \tilde{\Upsilon}_{\mathcal{U}\mathcal{L}}^\top$$

From these partial results and Assumption A.1, we can provide an approximated solution for $y(\tilde{\mathbf{x}})$ in (D.18). Specifically, Theorem D.7 shows the classification function of iLGC.

Theorem D.7 (Classification function of iLGC). *If Assumption A.1 holds, (D.18) is reduced to:*

$$y(\tilde{\mathbf{x}}) = \arg \max_{i \in \mathbb{N}_c} \left| \sum_{\mathbf{x}_k \in \tilde{\mathcal{N}}} \mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_k) \tilde{\mathbf{M}}_k^{-1/2} \sum_{t=1}^n \left(\tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1} \right)_{kt} \mathbf{F}_{ti} \right| \quad (\text{D.19})$$

Proof. From (D.18), we have:

$$\begin{aligned} y(\tilde{\mathbf{x}}) &= \arg \max_{i \in \mathbb{N}_c} \left[- \left(\tilde{\Upsilon}_{\mathcal{U}\mathcal{U}} - \tilde{\Upsilon}_{\mathcal{U}\mathcal{L}} \tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1} \tilde{\Upsilon}_{\mathcal{U}\mathcal{L}}^\top \right)^{-1} \tilde{\Upsilon}_{\mathcal{U}\mathcal{L}} \tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{F} \right]_i \\ &= \arg \max_{i \in \mathbb{N}_c} \left[- \tilde{\Phi}^{-1} \tilde{\Upsilon}_{\mathcal{U}\mathcal{L}} \tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{F} \right]_i \end{aligned}$$

Since $\tilde{\Phi} \in \mathbb{R}$, we obtain:

$$\begin{aligned} y(\tilde{\mathbf{x}}) &= \arg \max_{i \in \mathbb{N}_c} \left| \left(- \tilde{\Upsilon}_{\mathcal{U}\mathcal{L}} \tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1} \mathbf{F} \right)_i \right| \\ &= \arg \max_{i \in \mathbb{N}_c} \left| - \sum_{t=1}^n \sum_{k=1}^n \left(\tilde{\Upsilon}_{\mathcal{U}\mathcal{L}} \right)_k \left(\tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1} \right)_{kt} \mathbf{F}_{ti} \right| \\ &= \arg \max_{i \in \mathbb{N}_c} \left| \sum_{t=1}^n \sum_{k=1}^n \frac{\tilde{\mathbf{M}}_{n+1}^{-1/2} \mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_k) \tilde{\mathbf{N}}_k}{\tilde{\mathbf{M}}_k^{1/2}} \left(\tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1} \right)_{kt} \mathbf{F}_{ti} \right| \\ &= \arg \max_{i \in \mathbb{N}_c} \left| \sum_{\mathbf{x}_k \in \tilde{\mathcal{N}}} \mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_k) \tilde{\mathbf{M}}_k^{-1/2} \sum_{t=1}^n \left(\tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1} \right)_{kt} \mathbf{F}_{ti} \right| \\ &= \arg \max_{i \in \mathbb{N}_c} \left| \sum_{\mathbf{x}_k \in \tilde{\mathcal{N}}} \frac{\mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_k) \sum_{t=1}^n \left(\tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1} \right)_{kt} \mathbf{F}_{ti}}{\left(\frac{1}{\tilde{\mathbf{M}}_k^2} + \mathcal{K}(\mathbf{x}, \mathbf{x}_k) \right)^{1/2}} \right| \end{aligned}$$

□

Computing (D.19) directly may be costly because we need to compute $\tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1}$ for each unseen example $\tilde{\mathbf{x}}$. Assuming the sparsity assumption of the input graph, we make $\tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1} \approx \Upsilon^{-1}$. As the number of training examples grows, this approximation holds because the number of edges we create is much smaller than the number of edges of the input graph. Mathematically, $\tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1} \rightarrow \Upsilon^{-1}$ as $n \rightarrow +\infty$.

Therefore, we have:

$$\sum_{t=1}^n \left(\tilde{\Upsilon}_{\mathcal{L}\mathcal{L}}^{-1} \right)_{kt} \mathbf{F}_{ti} \approx \sum_{t=1}^n \left(\Upsilon^{-1} \right)_{kt} \mathbf{F}_{ti} = \Upsilon^{-1} \mathbf{F} = \frac{\mu}{1+\mu} \Upsilon^{-2} \mathbf{Y}$$

Let $\Omega := \Upsilon^{-2} \mathbf{Y} \in \mathbb{R}^{n \times c}$. Therefore, we rewrite (D.19) as follows:

$$y(\tilde{\mathbf{x}}) = \arg \max_{i \in \mathbb{N}_c} \left| \sum_{\mathbf{x}_k \in \tilde{\mathcal{N}}} \frac{\mathcal{K}(\tilde{\mathbf{x}}, \mathbf{x}_k) \Omega_{ki}}{\left(\frac{1}{\mathbf{M}_k^2} + \mathcal{K}(\mathbf{x}, \mathbf{x}_k) \right)^{1/2}} \right| \quad (\text{D.20})$$

Classification function in (D.20) is an approximated and effective solution for (D.19). Note that Ω and $\mathbf{M}$ can be computed during training stage. This decreases considerably the running time to classify out-of-sample examples in the input space $\mathbb{R}^d$.